(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
6 December 2001 (06.12.2001)

PCT

(10) International Publication Number
WO 01/92981 A2

(51) International Patent Classification[7]: G06F

(21) International Application Number: PCT/IL01/00487

(22) International Filing Date: 28 May 2001 (28.05.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

136414	28 May 2000 (28.05.2000)	IL
60/209,593	6 June 2000 (06.06.2000)	US
60,284,019	15 April 2001 (15.04.2001)	US

(71) Applicants and
(72) Inventors: **MAYER, Yaron** [IL/IL]; Ahad Haam Street 21, 92151 Jerusalem (IL). **DECHOVICH, Zak** [IL/IL]; Hasayeret Hayerushalmit 16/6, Pisgat Zeev, 97543 Jerusalem (IL).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: SYSTEM AND METHOD FOR COMPREHENSIVE GENERAL GENERIC PROTECTION FOR COMPUTERS AGAINST MALICIOUS PROGRAMS THAT MAY STEAL INFORMATION AND/OR CAUSE DAMAGES

(57) Abstract: Malicious software attacks (such as stealing data, changing data or destroying data) on personal computers and/or servers and/or other computerized gadgets (especially through the Internet) are becoming more and more common and more and more dangerous, causing damages of tens of billions of dollars each year. The state-of the-art solutions are inherently limited because they solve only a limited number of problems on the surface, instead of going deeply into the roots of the problem. The most common solutions are Anti-viruses and Network firewalls. Anti-viruses are limited because they can only detect known viruses or worms that have already been identified (usually after they have already attacked many computers). Network firewalls are based on packet filtering, which is limited in principle, since the rules of which packets to accept or not may contain for example subjective decisions based on trusting certain sites or certain applications. However, once security is breached for any reason, for example due to an error or intended deception, a hostile application may take over the computer or server or the entire network and create unlimited damages (directly or by opening the door to additional malicious applications). They are also not effective against security holes for example in browsers or e-mail programs or in the operating system itself. According to an article in ZDnet from Jan 24, 2001, security holes in critical applications are discovered so often that just keeping up with all the patches is impractical. Also, without proper generic protection for example against Trojan horses, which can identify any malicious program without prior knowledge about it, even VPNs (Virtual Private Networks) and other form of data encryption, including digital signatures, are not really safe because the info can be stolen before or below the encryption. The present invention creates a general generic comprehensive solution by going deeply into the roots of the problem. One of the biggest absurdities of the state-of-the-art situation is that by default programs are allowed to do whatever they like to other programs or to their data files or to critical files of the operating system, which is as absurd as letting a guest in a hotel bother any other guests as he pleases, steal their property or copy it or destroy it, destroy their rooms, etc., or for example have free access to the hotel's safe or electronic switchboard or phone or elevator control room. The present concept is based on automatic segregation between programs: It is like limiting each guest by default to his room and limiting by default his access to the Hotel's strategic resources, so that only by explicit permission each guest can get additional privileges.

WO 01/92981 A2

System and method for comprehensive general generic protection for computers against malicious programs that may steal information and/or cause damages.

Background of the invention

Field of the invention:

The present invention relates to security in computers (including personal computers, servers, or other computerized gadgets, as explained in the definitions] and more specifically to a powerful comprehensive generic Security System and method for computers, based on automatic segregation between programs.

Background

Malicious software attacks on personal computers and/or servers (especially through the Internet) are becoming more and more common and more and more dangerous. According to recent study by the American CSI research institute (The Computer Security Institute), during the last year alone approximately 50 percent of the largest American companies were attacked by at least one malicious Internet software, with an average damage of about 500,000 USD per attack.

As the last attacks by the "I LOVE YOU" virus and its derivatives demonstrate, which affected almost instantly tens of millions of computers and caused estimated damages of more than 10 billion dollars - the conventional anti-virus programs and their methods are inadequate to handle such threats because they are dependent on past familiar code patterns of known malicious software instead of trying to prevent in advance all kinds of attacks in principle. Such attacks are enabled because of an almost infinite number of loopholes and vulnerabilities in operating systems, by the fact that far too many processes occur under the surface without the user's knowledge, and by badly-engineered applications. These loopholes and vulnerabilities include for example:

1. The possibility of attempting to connect from the outside to the user's computer while the user is surfing on the net, without any warning to the user that such attempt is being made.

2. The readiness of certain applications to execute Macro commands or scripts or applets or other executable attachments from incoming e-mail messages or web pages without any warning and without asking for the user's authorization, and without checking what these executables or scripts try to do (if allowed to run).

3. The ability of applications to open a network connection to the outside without any warning or request of authorization from the user.

4. The ability of applications to perform extremely dangerous operations such as deleting or sabotaging multiple files or making changes to sensitive system areas or formatting entire drives without warning the user and requesting authorization from the user.

5. Lack of checks against string overflow or buffer overflow in some communication applications so that they can be crashed for example by large strings that contain malicious code that overwrites part of the original program's code and starts running instead of the original code.

Unless these vulnerabilities and loopholes are treated on the most thorough and basic level, and since the Internet keeps growing at an exponential rate and more and more businesses are becoming dependent on it - such attacks may increase in the near future to the point of almost unlimited damages to a very large percent of the computers that are connected to the Internet.

Other methods such as packet filtering are also limited in principle, since the rules of which packets to accept or not may contain for example subjective decisions based on trusting certain sites or certain applications. However, once security is breached for any reason, for example due to an error or intended deception, a hostile application may take over the computer or server or the entire network and create unlimited damages (directly or by opening the door to additional malicious applications), which until detected might be already too late to repair. For example, a self-resendable via e-mail macro-virus (such as "I LOVE YOU" and its derivatives and similar viruses) can arrive from your best and most trusted friends after their own computer has been compromised. Also, filtering for allowed types of protocols such as for example FTP versus SMTP and so on can be rendered useless by programs that encrypt or disguise a given protocol type to appear as another. Another major limitation of packet filtering is that it can't be relied upon to scan for stolen data in packets, since malicious applications can encrypt the data and/or disguise it to look like something else, so as to appear for example as a gif image.

Antiviruses and firewalls are also not effective against security holes for example in browsers or e-mail programs or in the operating system itself. According to an article in ZDnet from Jan 24, 2001, security holes in critical applications are discovered so often that just keeping up with all the patches is impractical. Also, without proper generic protection for example against Trojan horses, which can identify any malicious program without prior knowledge about it, even VPNs (Virtual Private Networks) and other form of data encryption, including digital signatures, are not really safe because the info can be stolen before or below the encryption.

Even attempts to monitor in some ways a certain group of specifically marked executables or applications are limited by nature, because the security breaches can come from many other directions. For example, A Trojan horse may already be lurking in the system for a long time and then suddenly create tremendous damage before being detected, or enter the system any time new applications are installed from any source.

On the other hand, attempts for example to disallow completely any script within e-mail attachments to execute creates too many restrictions and doesn't separate between safe scripts that the user may want to run and scripts that actually try to perform malicious acts.

Summary of the invention

The present invention is a novel concept which tries to go deeply into the roots of the causes of above described problems and thus to eliminate completely the above-described problems by creating what is to the best of our knowledge a most powerful, comprehensive, general and generic Security System for computers. This System and method protects computers (which may include personal computers, servers, and other devices or gadgets with one or more processor that can run programs, as explained below in the definitions) against all kinds of malicious programs that may steal information and/or cause damages including changes of data, deletion of data, interfering with function and so on (such as Viruses, Vandals, Trojan horses, Worms, Macro viruses and Malicious e-mails). The system and method can be used in many operating systems, such as various platforms of Microsoft Windows, Linux, Macintosh, or other operating systems, eventhough the preferred embodiments use mainly the terminology of Windows, which is the most common and familiar operating system.

The most important principles and objects of this protection system preferably include:

1. Giving the user more information about processes that would normally occur without his knowledge, thus decreasing substantially the chance that malicious software will be able to cheat the user.

2. Defining comprehensive yet parsimonic sets of rules of appropriate behavior of software so that the system can identify and intercept immediately programs that may be performing or trying to perform suspicious and/or detrimental and/or potentially dangerous activities or not behaving as usual.

3. Monitoring and intercepting and possibly logging all unauthorized and/or suspect activities in the computer and asking for authorization or guidance when required.

4. The above-described principles allow multiple safeguards against security threats, so that malicious applications will usually have to break multiple security rules in order to do such things as stealing data, damaging data or propagating themselves, and thus the chance for catching them is much larger.

5. Even if the user allows a certain application to launch another application, the newly launched application or applications are again subjected in turn to all the same monitoring and rules as any other application, so that the scanning for breach of security rules continues to apply at all stages.

6. Since the possibility of encryption by malicious programs which try to steal and send data over communication channels makes it impossible to make sure by monitoring the information flow itself that data is not being stolen – therefore the system relies mainly on allowing the user maximum control over which applications can access which data, which applications

are authorized to access which communication channels, and how much data is actually being sent.

The above protection system is preferably comprised of the following main elements:

1. A monitoring and capturing system, which constantly monitors the security-sensitive elements of the computer system, and most importantly all the relevant peripheral device activities, and especially those related to storage devices (and especially the Hard disk or hard disks) and communication devices (network cards, modem, etc.) and can detect and intercept immediately any suspicious or dangerous behavior.

2. The security rules and a database (or databases) for storing the default rules, a set of pre-distribution acquired rules that are good for most users of the selected operating system, the acquired additional user-defined rules, and the statistics of normal or reasonable behavior of programs, which is continuously learned during the operation of the system. This database area, which contains also all the authorizations and optionally (preferably) a log of all the questions that the Security System asked the user and his replies (kept at least for a certain period), and when needed, also a log of suspicious activities detected (kept at least for a certain period) and may contain also definable additional logs. The database is preferably encrypted and is considered a constantly monitored high-security protected and preferably backed-up area as defined below in the detailed description

3. A user interface, which interacts with the user in order to learn acceptable behavior patterns, warn the user of perceived dangers when needed, and ask for the user's authorization when needed. Preferably it also allows the user to view statistics of behavior of important programs or groups of programs and especially programs that are allowed to access communication channels, especially in what is related to sending and receiving data over the communication lines, such as since the beginning of the current Internet session or for a certain time period. Preferably, this may also include information such as what protocols were used, etc. The user may also view or modify directly the database of authorizations.

The main functions performed by Security System:

The main logic behind the rules that define appropriate versus suspect behavior is preventing as much as possible all the elements and activities that are required by malicious programs in order to be able steal any data or do any damage or propagate themselves. The Security System uses a set of heuristics and basic rules for defining suspicious or potentially dangerous activities that are automatically fit for most users. By using the general default rules and adding to them statistical analysis of normal system and applications behavior and what is learned from the user's responses to authorization requests, the Security System quickly learns what is considered reasonable or well-behaved behavior of programs on the user's personal computer or server. Preferably, some of the learning is performed in advance for each operating system and is included in the distribution database, so that the system that is installed

by the user has already learned various rules that are relevant by default to most users of that operating system. The security rules and functions performed by the Security System preferably include:

a. Constantly monitoring the security-sensitive elements of the computer system, including all relevant peripheral device activities, and especially storage devices and communication devices, and detecting and selectively intercepting any suspicious or dangerous behavior and acting upon it in accordance with the default and acquired sets of security rules,
b. Default segregation of programs into their natural environments, as defined below in the detailed description,
c. Warning the user and request for authorization for security-sensitive activities and especially any first-time attempts to access communication channels,
d. Preferably constant and stricter monitoring and protection of areas defined in the various rule sets as higher security areas on the storage media, as defined below in the detailed description,
e. Interception and more explicit warning of the user about potentially highly dangerous activities,
f. Warning the user about significant statistical deviations from normal behaviors of applications and operating system and especially as relates to suddenly sending out large amounts of data,
g. Allowing the User to request automatic immediate interception and/or warning the user of any attempts of external programs from the network to connect to the user's computer through the communication channels,
h. Preferably allowing the User to request enforcing of general limitations on the communication ports allowed to be opened and optionally also limitations on types of protocols allowed,
i. Monitoring and intercepting as much as possible all attempts of applications to gain direct port accesses to security sensitive devices and especially the storage media and the communication channels.

Therefore, the present invention offers the following main advantages over the prior art:

1. It enables generic detection and interception of all kinds and variations of viruses, Trojan horses, worms, E-mail macro viruses and other vandals even when these are completely new and not similar to other vandals encountered before. Therefore, it can also detect and intercept first strike attacks, instead of waiting for a cure after the damage has already been done to tens of millions of computers.
2. It is not dependent on constant updates of virus knowledge bases, unlike normal anti virus systems.
3. It is not dependent on inherently limited methods, such as packet filtering.
4. It offers multiple safeguards against various threats, so that a malicious program will typically have to break multiple security rules, and thus have a much higher chance for being caught. It gives the user more knowledge of

what is happening in his system and therefore reduces very significantly the chance of the user being cheated by malicious applications.

5. It is much more comprehensive than other solutions and may even ultimately catch and intercept backdoors that might exist in the operating system itself. Also, it is not dependent on marking a limited group of applications for being monitored, so that all applications are checked, no matter how they are introduced to the system, including if they were there even before the Security System was installed.
6. It is very parsimonic in nature, so that it doesn't need specific knowledge about the exact nature of specific programs such as various browsers or e-mail programs, and therefore also no updates are needed when the user downloads for example new versions or kinds of Internet applications.
7. Malicious behaviors of programs can be detected and intercepted even if they don't display viral or worm-like behavior at all, for example if a screen saver starts to steal data and send it out over communication lines even if it does not show any attempts to spread itself or to modify system areas.
8. Even systems protected by tight encryption policies, such as online banks, are not really safe without our Security System, because malicious software, such as for example the Subseven Trojan, can render the encryption useless by sending outside information on everything that is going on in the system at various levels.

Definitions

Many times for simplicity of understanding we use terms that are most commonly used within Microsoft Windows environment (which is the most common operating system for personal computers), so it should be kept in mind that in other operation systems such as Linux or Macintosh some of these might have different names, somewhat different implementations, etc., although the principle is the same.

As used throughout the present specifications and claims, the following words have the indicated meanings:

"Program", "executable" or "application" is any file or area in memory that contains executable commands, such as .exe or .com files, batch files, various Macro files, etc.

"Macro" is an executable written usually in a scripting language and executed by a complex application, such as Microsoft's Outlook or Word.

"DLL" is a dynamic link library. This term is common for example in all versions of the Windows operating system. In other operating systems it might have different names but the principle is the same. In general it is a term for a set of routines that can be called from executables, loaded and linked into them during run time.

"Device driver" or "Driver" is a software component that allows an operating system to communicate with one or more specific hardware devices attached to a computer, such as a hard disk controller, network card or display card.

"OS" or "operating system" is software responsible for controlling the allocation and usage of computer hardware resources such as memory, CPU time, disk space, and peripheral hardware devices.

"IRQ" or "Interrupt request line" is a hardware line over which a hardware device, such as an input/output port, keyboard, or disk drive, can send interrupt requests to the central processing unit (CPU). Interrupt request lines are built into the computer's internal hardware and are assigned different levels of priority so that the CPU can determine the sources and relative importance of incoming service requests.

"User" or "users" as used throughout the text are always meant interchangeably to be either user or users. The user or users can be for example the individual user of a computer or computers or a corporation or organization that uses the computers. Therefore, preferably various types of authorizations for example can be given either by the individual user of the computer or for example by the security administrator of the company, or any combination of these. For example some companies might want to give full authority on critical issues only to the system administrator, while others might want to let the employees or certain employees have much more direct control.

"User authorization" as used throughout the text can include also of course additional guidance and options.

"Database" or "Databases" as used throughout the text are always meant interchangeably to be either database or databases.

"Network" as used throughout the text is always interchangeable as either network or networks and represents a connection from a computer (as defined) by any way to one or more computers or any other compatible communication device.

"File" is one or more areas on one or more disks and may have a definition in the FAT that may be represented as a name, directory, etc. and may have other parameters.

"Registry" is one or more files that may contain operating system and other program settings and mainly managed by the operating system.

"Computer" can refer to a personal computer or workstation or server, or any automated device or gadget with one or more processor or CPU, capable of more than simple arithmetic functions. This includes for example also cellular phones and portable computing devices such as palm pilot. This includes also, for example, computers in cars, which may for example become very important as cars become more automated or even capable of automatic driving, since if hackers are able to damage them for example by Internet or satellite connection, it might even cause life-threatening malfunctions. Other examples can be computers in satellites (In which case, user authorization, when needed, preferably should be requested remotely by encrypted communication with user remote verification), sensitive computer systems in airplanes, etc. So, eventhough we give the examples usually from a PC and Windows perspective, similar principles can be applied also to palm devices, cellular phones, and other types of computerized devices. Also, "computer" or "computers" as used throughout the text are always meant interchangeably to be either computer or

computers. Therefore, whenever the word "computer" or "computers" is used throughout the text of this patent, including the claims, it can mean any of the above defined devices.

"Server" is a computer on a network that is running software that provides data and services to clients over the network. The term server can also apply to a software process, such as an Automation server, that similarly sends information to clients and that appears on the same computer as a client process, or even within the same application.

"Kernel" is the portion of the operating system that manages and controls access to hardware resources. It performs for example: thread scheduling and dispatching, interrupt and exception handling, and multiprocessor synchronization.

"DMA" is Direct Memory Access.

"Image Loading" as used throughout the text refers to an executable code that is being loaded for execution or unloaded/terminated.

"Hooked function" as used throughout the text refers to an executable filtering code placed between the calling code and called function and thus has the ability to monitor and/or intercept and/or redefine the function that is being hooked.

Brief description of the drawings

Fig. 1 shows the preferred main elements of the Security System within a typical structure of an operating system in a computer, with some of the hooked peripheral device drivers, especially those related to storage devices and network devices, and preferable places and ways that the various parts of the Security System are coupled to and interact with the above typical structure.

Fig. 1b shows in more detail a preferred way of interaction between Security System parts with an emphasis on the user interface and a preferred process of permission granting.

Fig. 2 shows in more detail a flow diagram of a preferred way the monitoring and capturing system interacts, monitors, checks and authorizes file hooked functions of the computer's operating system that may be preformed by an application.

Fig. 3 shows in more detail a flow diagram of a preferred way the monitoring and capturing system interacts, monitors, checks and authorizes network hooked functions of the computer's operating system that may be preformed by an application.

Fig. 4 shows in more detail a flow diagram of a preferred way the monitoring and capturing system interacts, monitors, checks and authorizes registry hooked functions of the computer's operating system that may be preformed by an application.

Fig. 5 shows what preferably happens when executable files are being loaded for execution.

Fig. 6 shows in more detail a flow diagram of a preferred way the monitoring and capturing system interacts, monitors, checks and authorizes memory related functions of the computer's operating system that may be preformed by an application.

Fig. 7 shows in more detail a flow diagram of preferred main parts and methods of the Security System database, permission and analysis processes.

Fig. 8 shows in more detail preferred interfaces and operation of a possible variation of using additional hardware, which monitors hardware accesses on the computer's data bus and has a 2-way interface with the Security System's software.

Fig. 9 shows in more detail an overview of a preferable self-preservation method.

Fig. 10 shows in more detail a flow diagram of a preferred method of interception process.

Fig 11 is a graphic illustration of a preferable way in which processes may be segregated and controlled.

Fig 12 is a visual illustration of a more extreme implementation of keeping each program in a 'Bubble' of virtual environment.

Fig 13 is a visual illustration of a preferable configuration of connecting computers in an organization to Internet for example through the system administrator's computer.

Detailed description of the preferred embodiments

All of descriptions in this and other sections are intended to be illustrative and not limiting.

Referring to Fig. 1, we show the preferred main elements of the Security System (100) within a typical structure of an operating system (101) in a computer (which can be for example a server, a personal computer, or other computerized gadgets or devices as explain in the definitions), with some of the hooked peripheral device drivers, especially those related to storage devices (110) and communication devices (111), and preferable places and ways that the various parts of the Security System (100) are coupled to and interact with the above typical structure. The entire system and method can be regarded also as a virtual machine that performs the described functions.

The Security System is preferably comprised of the following main elements:

a. A monitoring and capturing system (102), which constantly monitors the security-sensitive elements of the computer system, and especially all the relevant peripheral device activities and especially storage devices (110)(especially the Hard disk or hard disks) and communication devices (111) (network cards, modem, etc.) and can detect and intercept any suspicious and/or detrimental and/or potentially dangerous behaviors. This element of the Security System installs at least some parts of itself as much as possible in the kernel of the operating system (104), and other parts replace

various OS files, such as certain drives, device drivers, DLLs, etc. in order to hook various vital functions. The monitoring and intercepting system is defined in more detail in subsequent figures.

b. Security rules (740) and a database or databases (700) for storing the default rules (74X-C), a set of pre-distribution acquired rules (74X-B) that are good for most users of the selected operating system, the acquired additional user-defined rules (74X-A), and the statistics (751) of normal or reasonable behavior of programs, which is continuously learned during the operation of the system. Preferably this database (700) contains in addition to all the authorizations, an optional log (770) of all the questions that the Security System asked the user and his replies (kept at least for a certain period), and when needed, also a log (770) of suspicious activities detected (kept at least for a certain period) and may contain also definable additional logs. The database (700) is preferably encrypted and is considered a constantly monitored high-security protected and preferably backed-up area as defined below. Therefore, all accesses to the database are supervised by the monitoring and capturing system as explained in more detail in fig. 11.

c. A user interface (103), which interacts with the user in order to learn acceptable behavior patterns, warn the user of all perceived dangers and ask for user's authorization or guidance when required. Preferably it also allows the user to view statistics and behavior logs of any program or groups of programs in the computer that the user defines or programs that are considered strategically important such as programs that are allowed to access communication channels. For example one of the activities that is being statistically checked and analyzed is the amount and the data that is being send or received, the protocol that is being used, address etc. The user may also view or modify directly the database of authorizations. Preferably, the user may also choose the security software's tightness in a certain range of severity.

The Security System may also include (as another possible variation) an optional hardware element (800) shown in more detail in Fig. 8, which can alert the Security System's software to any events where access has been made to the security-sensitive ports (803) and/or memory (801) without an apparent corresponding event on the system level as monitored by said Security System's software.

Further referring to Fig. 1, preferably the main rules and functions performed by the Security System are:

1. By default, each program (software application) is allowed to access (read, write, execute, create, delete, etc.) files only within its natural environment (which is mainly the directory in which it is installed, its sub-directories, and (for reading only) non-strategic shared files). This way, even applications that are run within other applications, such as Java or Active-X within browsers, still have to conform to the security rules together with the browser itself. Preferably, the user may also ask in advance to protect and monitor only certain directories (or groups of directories), but by default all directories

are monitored. If the program is attempting to be installed in the root of any drive, preferably the user interface part (103) of the Security System warns the user about it, and if he allows it, then the natural environment of such programs is limited only to the root of that drive and does not include its sub-directories, otherwise the segregation to branches would be meaningless in this cases. Similarly, the Security System constantly monitors, intercepts, and warns the user of any attempts by programs to access the storage devices (110) through direct I/O, since that could render meaningless the segregation rules. (This can be accomplished for example by putting the Security System in ring 0 – Using Intel architecture terms). This can be viewed either as a segregation of programs or of processes, however it is more preferable to implement it according to program files, since for example two or more copies in memory of Netscape will still typically have the same privileges and definitions. On the other hand, if different threads are run by some programs, for example Java or Javascript by Netscape, they either can be treated separately as processes, or they can be identified separately anyway for example by the file from which the DLL originates, etc. Another way to explain it, which also shows the absurdity of the state-of-the-art situation, is to compare the computer to a hotel. Allowing a program to do whatever it likes to other programs or to their data files or to critical files of the operating system is as absurd as letting a guest in a hotel bother any other guests as he pleases, steal their property or copy it or destroy it, destroy their rooms, etc., or for example have free access to the hotel's safe or electronic switchboard or elevator control room, or phone. The present concept is like limiting each guest by default to his room and limiting by default his access to the Hotel's strategic resources, so that only by explicit permission each guest can get additional privileges.

2. By default, no program is allowed without permission to access and especially to modify or replace sensitive areas or files (as defined below) or device drivers in the storage media (and preferably, to the extent possible, sometimes also in the computer's RAM (112)), which are regarded by the Security System as High security areas, such as critical operating-system files, registries, INI files, important DLL (Dynamic Link Libraries) files and communication-related files (such as Winsock, etc.), the boot sector, the FAT, autoexec or configuration files, the initialization areas of the operating system, the Windows Startup directory, the BIOS, user defined high security files or directories, system files that contain lists of URLs from which drivers can be automatically downloaded without asking the user (such as in Windows 2000), all the executable and data files and databases related to the Security System itself, or anything else that may prevent the Security System from continuing functioning properly or initializing properly after the next boot. Similarly, the Security System preferably constantly monitors attempts by various programs to access directly the area of the hard disk used by the

operating system for the swap files, since that could also allow various security breaches, such as for example replacing critical DLLs with malicious DLLs while they are cached on the disk during virtual memory swapping. In addition to this, the system may preferably to the extent possible also protect (600) some RAM (112) areas if they are not adequately protected by the computer's operating system (101). For example, there might be a vulnerability that enables applications to access a shared memory area called "System internal object name space" and change the names of DLLs, thus replacing them with references to malicious DLLs. In addition to this, the Security System preferably makes sure (600) that it will not be thrown out of the RAM by other applications that might try to neutralize it, for example by checking all the time that it is not thrown out of the DDB (Device Descriptor Block) by other applications and putting itself all the time in the first place there, and/or by the methods described in Fig. 9 about self-preservation. Preferably, unless it is sufficiently done by the operating system itself, to the extent possible, the Security system also prevents programs from accessing also in memory the code or data of other programs or their drivers or DLLs, etc. (unless given explicit permission to do so).

3. In addition to this, as much as possible, preferably most of the high security areas described above in clause 2 are monitored regularly for signs of suspicious changes by means of a hidden fingerprint for each such file which will cease fitting the file if an unauthorized change has occurred, and preferably there are also additional hidden encrypted and regularly refreshed backups of important areas that can be used to restore them in case of damages to them.

4. Any program that tries to access (such as send, receive, listen, connect etc.) communication channels (111), including IP address, port and protocol (mainly win-sockets and network shared device drivers (300)) needs to get permission from the user (unless it has already been given this privilege). Based on this monitoring, the user is warned and is asked for authorization (for any previously unauthorized connection), inbound or outbound, including any attempts from programs or hackers from the network (120) to connect to the user's computer, and the Security System may also trace-route such attempts on the net (120) in order to find the source of the attack. Preferably, when the Security System asks the user for example if to allow a certain application to access communication channels, it shows additional relevant data apart from the application's name, such as, for example, the full path of where the executable is installed, its size, its date, and/or details such as for example CRC, memory segments, or other identifiers, in order to reduce the chance that some hostile application might for example install itself under some directory and name itself netscape.exe and thus be given inadvertently by the user access to the web. Similarly, preferably, if another application with the same or similar name is

already listed in the Security System's Database, it preferably warns the user about this, in order to further avoid confusion. If the user is for example an organization and the organization wants for example to allow the system administrator to control which applications have access to the web, then for example each time an employee working with a certain computer allows a certain application to access the web, then preferably this can be permitted only if it fits the definitions allowed by the administrator, preferably using various identification marks to make sure that it is indeed an allowed application and not some other executable with the same name. This can be accomplished in a number of possible ways: For example the administrator can define allowed applications with their identification marks and broadcast this once in a while to all the computers in the organizations, and the Security system will allow access to communication channels only to applications that comply with these definitions (preferably these definitions are password-protected and also reside in an area regarded as a high-security area). Another possible variation is that various requests for authorizations (preferably including various identification marks of the applications) are broadcast by the security system directly to the administrator without even asking the employee and preferably remain blocked until authorization can be given by him. Another possible variation is for example that new authorizations given to applications by the employee (or at least authorizations on important issues) are broadcast by the security system also to the administrator, and allowed only if he OKs them. Another possible variation is for example that, at least for certain authorizations, the user has to call the administrator, and only he can authorize them for example with a password. Another possible variation is for example that applications that are allowed to access the web and/or other communication channels reside only in one (or more) computers in the network and the other computers can access them for example only by limited access through local-area network. Various combinations of these and other solutions are also possible. Also, preferably the Security System allows the user to define general limitations on the communication channels (111) allowed to be opened and optionally also limitations on types of protocols allowed, which is especially useful in cases where the computer is being used as a server, since in such cases the computer will run most of the time unattended by the user, or for example if the user wants to block automatically all incoming communication attempts and just log them. Additionally, due to the nature of E-mail Macro-viruses, as added security measures, the system preferably constantly monitors the communication channels for outgoing E-mail messages and asks the user for confirmation any time that one or more e-mail messages are being sent out by any program (even authorized programs) or at least and especially when multiple E-mails are being sent out consecutively. Preferably, the Security System also learns by this process various characteristics of the way the user is normally sending e-mail messages, so that whenever sudden unusual

characteristics are apparent, preferably a special interception and warning can be issued. For example when sending e-mail normally through a program like outlook express, the relevant MAPI functions will be called differently and/or other processes may happen differently than for example when sending e-mail from a Visual Basic Script executed by outlook express. In addition to this, since programs that are allowed to access the communication lines (and especially browsers and e-mail programs) are usually a crucial link in Internet related attacks, preferably such programs are always monitored more thoroughly by the Security System, and therefore regarding such programs preferably the user may not tell the Security System to stop asking about various behaviors. Examples of said communication channels in terms of hardware can be the modem, Ethernet card(s), or even the USB (Universal Serial Bus), which can also be used for example for ADSL connection, or any other device that exists or might exist in the future which might be used for communicating data in and out of the computer. This comprehensive covering of all possible communication channels is extremely important, since otherwise the whole security system might be rendered useless. Examples of said communication channels in terms of software can be any of the system functions that can access any of the said hardware devices that can be used for communication, including for example TAPI functions, which can use the modem for sending Faxes, since, otherwise, a malicious application might for example turn off the internal loudspeaker of the modem and dial out and send out stolen data as a Fax. This applies also for example to any access to wireless channels, such as for example Bluetooth or infra-red, since this also can be used for sending data to the computer or stealing data from it.

5. Preferably, the monitoring & capturing system (102) conducts constant statistical analysis of various events in the computer in order to learn about normal behavior and identify significant deviations from the normal behavior (such as sending out significantly more data than usual, accessing more files than usual, etc.). Preferably, especially the programs that have been authorized for use of the communication channels (111) are constantly statistically analyzed and monitored for suspicious deviations from their normal statistical patterns of behavior, so that if such a program for example suddenly starts to access significantly more files than usual or scan large areas of the disk (even if it has been allowed by the user to access areas outside its natural environment) or starts to send out unusual amount of data, it is immediately intercepted and the user warned and asked for authorization. This is important also in cases that programs start to act strange due to their being changed while already loaded in memory, for example because of some hardware failure or because they were crashed by string overflow, etc.

6. The Security System monitors as much as possible all attempts of software applications to gain direct port accesses to security sensitive devices (such as the modem and network cards (111), hard disk controller, etc.), or to bypass the win-socket drivers, since such access could bypass the operating system. Windows NT for example allows only drivers that are installed in ring 0 to access such ports directly, so ordinary applications are automatically prevented from doing this, but some other versions of windows do not enforce this limitation. Therefore, the Security System tries to enforce this as much as possible even on systems were it is not enforced.

7. During its own installation, the Security System preferably performs various checks if various crucial system files are suspicious of being infected already, and in that case might for example recommend to the user to reinstall the operating system before trying to install again the security software.

8. In order to solve the security problems created by the existence of writeable shared directories such as the windows temp area, the Security System preferably implements also a new concept: Virtual Shared Directories. This way, each time an executable tries to access such a shared directory, it will be preferably given the illusion that it has accessed it, but in reality, each such executable will be preferably redirected to a separate private sub-directory which only it can access. Similarly, when executables are accessing shared keys in the registry, the Security System preferably implements also a Virtual Shared-Keys system such as registered components, etc., so that again, preferably the executables are given the illusion that they have accessed the shared keys, but preferably they are in practice being redirected each to its individual private file of relevant registry keys. This, in combination with the other rules/functions, and especially rule no.1 (about the automatic segregation), can also be described in other words as a system of multiple automatic sandboxes, or a system in which each program is limited to its own virtual computer.

9. To the extent possible, the Security System preferably also tries to push the operating system or at least parts of it, to the extent possible, from processor ring 0 (privileged) to ring 1 (less privileged), preferably with the aid of an additional component that converts all the needed functions to run in ring 1 instead of ring 0. This is an additional way to block any hidden doors that might exist in the operating system. And of course it allows easier control over the access to system resources. Although these rings are concepts in Intel processors, similar rings or concepts might exist also in other processors.

Among other things, this system and method are important for example for the prevention of theft of highly sensitive codes, such as private encryption keys or credit card details. This is important because in the USA a recent legislation regards digital

signatures as no less obligating than handwritten signatures, and in other countries there are similar legislations in process. One of the biggest service suppliers in this area brags that it could take almost infinite time to break the private keys in these digital signatures, but ignores the simple fact that there is no need to break the keys since it is much easier to steal them, for example by a Trojan horse arriving by e-mail or through a web page by exploiting various loopholes in browsers or e-mail programs. Since such a signature can be compelling in any kind of contract, including wills and huge real estate deals especially in places where it comes with a non-repudiation policy, it is clear that the damage from stolen keys can be enormous. This is especially dangerous since many times such private keys are generated and stored for example by the browsers. By enforcing the rules, such as the automatic segregation rules and requesting authorization from the user about any access to any communication channels, such theft can be thoroughly avoided. However, in the cases where the keys are generated or stored by the browsers, preferably additional rules are used in order to identify the directories where these keys are held, otherwise accessing the keys by the browser would be within the browser's default authorization. Also, preferably, in this case the Security System also learns various characteristics of the way the user is normally accessing the keys, so that when sudden unusual characteristics are apparent, preferably a special interception and warning can be issued. Even if hardware cards, such as smart cards, are used for storing the encryption keys, these keys might still be stolen by Trojans for example by overwriting parts of the programs that access these cards or by monitoring the data in memory while it is being generated by these programs. In cellular phones, for example, eventhough they usually don't have yet sophisticated or sensitive operating systems and file systems compared to Windows, for example, and the operating system is usually EPROMM based, still at least some of the principles of the present system and method can be applied, such as:

1. The self-defense principles, such as requiring authorization to modify the BIOS's EPROMM and such as outlined for example in Fig. 9, and protecting the system-critical areas, are easier to implement, since the entire operating system and the security system may be on EPROMM or similar non-easily modifiable memory. So, for example, any attempt to modify any EPROMM data needs explicit permission from the user.
2. The RAM memory used for processing data operations is preferably monitored against hostile activities.
3. Since cellular phones and other mobile devices will be used more and more for business transactions, such as buying from e-commerce sites, transferring funds, stockmarket instructions, etc., the security of sensitive codes such as credit card data and especially private encryption keys is extremely important. So, for example, any attempt to access the private encryption Keys in any way preferably needs explicit permission from the user. Preferably in these cases the user is also asked for a password, which helps for example in cases where the phone is stolen.
4. Any attempt to automatically dial-out or automatically answer incoming calls preferably needs explicit permission from the user, especially if multiple automatic dials are attempted. This prevents any viruses from causing the phone to automatically send messages to various places, or from becoming for example a spying device, recording what is going on in the room and sending it out without the user's knowledge.

5. In case of constant open Internet connection, expected for example in the 3rd generation cellular phones, like in the PC example, preferably no program can access the web without prior user permission and no connection initiated from the outside can come-in without user permission.
6. Any unauthorized access to additional communication channels, such as Bluetooth devices also is preferably blocked or has to be authorized by the user.
7. As cellular or mobile or other phones become more sophisticated and computerized or for example integrated with palm devices, they may contain more and more features typical of ordinary computers and operating systems, and then more features of the present system and method may be needed and implemented.

More technical details are described in the following figures.

Fig 1b shows in more detail a preferred interaction between Security System parts with an emphasis on the user interface (preferably graphic user interface) and a preferred process of permission granting. As soon as a program (executable) tries to perform what is considered by the Security System as suspect or potentially dangerous activity (such as exceeding the natural environment of the program or trying to access the communication channels), the monitoring and intercepting system (102) immediately stops the program (1002) and asks the user for authorization, and if the user is absent, for example in case of protecting a server, suspect activities may be either blocked until the user comes back and/or logged (770), and such decisions are made either according to the various sets of security rules (740) and the nature of the suspect or dangerous activity, or by user definition. Preferably, for non-highly dangerous activities (1106), the Security System gives the user options such as for example to abort the offending program immediately, allow only this time, disallow but let the program go on, allow always from now on or until a certain event, stop asking completely about similar breaches for this program, or stop asking completely about similar breaches for all programs in this directory and it's sub-directories. If the suspect activity is related to files, the Security Systems preferably asks also if the permission is given only for reading of data or also for modifying data, etc. If the suspect activity is related to communication channels, the system preferably also allows the user to specify which channels to allow the application to use and what related activities to allow. The examples (in all of the figures) are intended to be illustrative and not limiting. Preferably, in order to avoid careless responses by the user, the user is always asked for authorization in such ways that responding without paying attention will always default to the least dangerous options. Preferably, for highly dangerous activities (1108), such as formatting a drive, mass deletion of files, changing hard disk partition information, changing boot area information, installing drivers in levels close to the kernel of the operating system, accessing the defined high-security areas, or modifying executables that reside outside the natural environment of the offending executable programs (such as exe and com files, batch files, DLLs, MS-DOC, MS-XLS files, or any other file that might contain executable commands), renaming them, creating new executables, or changing the linking of files types with applications that will be run when clicking on them, etc. - the user is warned more explicitly (preferably also with an explanation about the possible implications and causes) and preferably asked to repeat the authorization at least twice. Preferably, this is applied in all cases – even for programs that were allowed by the user to exceed their natural environment. Preferably, when the user is asked for authorization, the security system

also makes sure that no other programs can enter false answers as if they were entered by the user through the keyboard or the mouse or any other input device, for example by preventing other programs (except the allowed relevant input device drivers) from adding data for example to the buffer of typed keys in memory and the buffer of mouse events, or for example by using the hooking of all keyboard access and all mouse events to make sure that whatever is read for example from the keyboard or mouse is identical to what is in their event buffers or using only the commands that come directly through these hooked functions. Another possible variation of this is that the Security System freezes all other processes while it is waiting for the user's reply, for example at least for highly dangerous activities. Another possible variation of this is that the Security System plants its own keyboard and mouse drivers instead of those normally in use, however this could be problematic when a non-standard keyboard or mouse is used. Another possible variation of this is to use for example a smarter keyboard and/or mouse which uses also encryption preferably with a date & time stamp, like in the communication with the administrator's computer, as explained below. In addition to this, preferably the Security System also controls access to events and to objects (such as for example the edit box) and to the memory of programs such as for example shell32.dll,user32.dll & gdi32.dll (which are related to the Windows user interface, for example when using the standard open file dialogue box), so that programs don't create false events (such as for example pressing the OK button even though it hasn't really been pressed) or for example alter by direct memory access the content of the input line that contains the file name. Preferably these or similar methods can be applied also for example in systems that allow voice commands. Another possible variation of this is to ask the user also for password for at least some of these authorizations, such as for highly dangerous activities, which is good also in order to decrease the chance that the authorization will be given by someone else for example while the user is temporarily away. Of course, various combinations of these methods can also be used. Preferably, In addition to this, Like in the examples given in the reference to Fig. 4, the Security System also preferably identifies if the user or the application initiated a potential security-risk command, such as for example accessing a file outside the natural environment of the program for a program that still does not have that privilege, and so can for example allow more flexibility and less limitations (or even no limitations) if the command was initiated directly by the user than if it was initiated by the application. This can save the need to ask the user for confirmation in cases where he himself directly initiated the command, for example when it is not a highly dangerous activity. Again, in order to make this reliable, the Security System preferably prevents applications from creating the false impression as if the user for example typed something on the keyboard and thus initiated the command, preferably by use of any of the ways described above. Preferably, additional definitions of highly dangerous activities may be easily supplied as an update. However, in order to avoid issuing multiple warnings for example while a program is installing itself, preferably the Security System can handle it smartly as a continuous action within the same context. Also, when a new program installs itself, preferably the security system records which files are created by it, in order to be able to identify more easily its associated files even when they are in other areas. Preferably the Security System also analyses during the installation the imported functions in shared DLLs of the program in order to try to anticipate the behavior of the program and its needs. So preferably the Security System is installed as soon as possible after the operating system is installed, before other applications. (However, as explained above, the Security System can work also for applications installed before it). Also, in order to make this more efficient in organizations, preferably one computer can be used for example for learning all of the

segregation rules and various environment parameters for each program and this knowledge can be transferred to all the other computers in the organization, regardless of the order in which the applications are installed in the other computers. Like the examples given in function number 4 in the reference to Fig. 1, If the user is for example an organization and the organization wants for example to allow the system administrator to control some or all of the important authorizations, or all of the authorizations, or for example all the potentially highly dangerous activities, then preferably various or all requests for authorization can be for example referred by the Security system directly to the system administrator instead of or in addition to asking the employee that works with the computer, or for example automatically blocked unless they fit with pre-defined permissions by the administrator (that can preferably be easily updated by him whenever needed), by methods like those described in the relevant examples given in function 4. Also, preferably various information such as for example parameters or suspect behaviors learned on one or more computers can be transferred to other computers, preferably only after authorization by the administrator. Preferably all communications with this authority (such as the administrator's computer) are secure and encrypted and preferably include also an exact time and date stamp, in order to prevent malicious programs for example from trying to send false authorizations or reuse old authentic authorizations for generating false authorizations. Also, preferably this communication uses special protocols of the security system instead of the normal network device drivers and protocols of the operating system. This can enable also in practice general policy enforcement, so that the organization can decide and enforce very easily for example that only a certain set of programs may be run on all or on certain computers, or only certain actions are allowed on all or on certain computers, etc. These various options can be regarded as various possible embodiments. Some of them can be made available for example as separate products, or for example as various options within the same product. Preferably on each computer in an organization the level of control given to the employee versus the control for example by the system administrator can preferably be set independently for each computer in the organization.

A preferable way of viewing and/or modifying the database of authorizations is for example in the form of a table which lists the names and preferably various identification marks of applications allowed to access communication channels (and preferably a list of which channels), or to exceed their natural environments, or to have any other privileges which normal applications do not have by default, and lists which such privileges they have been given. Some activities might remain unallowed to any applications, such as for example trapping the keyboard device in order to catch keystrokes. Preferably this table includes also various statistical data about the behavior of each program, as explained before. In an organization where most control is in the hands of the system administrator, preferably the security system installed on each computer still runs a similar table and maintains a similar database, however the system can limit what the employee can change without the administrator's authorization. In such an organization, preferably on the administrator's computer this table contains also additional information and controls, such as for example the list of computers connected to the system within the organization, preferably with a unique identifier to each computer, and preferably with additional statistical information on the behavior of each computer in the list, so that preferably the system can alert the administrator for example whenever a computer in the system starts to deviate significantly from its normal behavior, such as unusual disk activity or unusual communications activity. Such data is preferably also logged. Preferably the communication between the administrator's computer and the employees'

computers is encrypted and secure. Preferably, in addition to this, the Security System on the administrator's computer constantly sends short communications at short intervals to the other computers in the system in order to be able to notice quickly for example if the Security System on any computer has been damaged or disabled. Preferably, this short communication can contain for example special codes with different keys for each computer, so that only an active Security System can respond to it properly, and so that a different response will come from a working computer where the Security System has been disabled or is not working properly (including for example if the computer was booted from a diskette instead of the hard disk), and no response from a computer that is turned off for example. Another possible variation is that preferably, in addition to this, if the computers in the organization are configured to access the web for example only through one (or more) central gateway computer (which might be the administrator's computer, or a separate computer), as shown in Fig. 13, so that for example each normal computer in the system does not have an independent modem and only has for example a network card), this might be used as an additional control for catching backdoors that might exist for example even in the operating system itself: In this case, preferably all communications are routed also through the administrator's computer, and the Security System on each computer preferably reports to the Security System on the administrator's computer all the time or at preferably short intervals for example how much data it has allowed to be sent out from the computer's communication channels, so that the Security System on the administrator's computer can preferably notice and intercept immediately or after a short interval communication attempts from computers where the amount of actual communication does not fit the amount reported by the Security System of that computer. In order to find how much data has been actually sent by each computer, the Security System on the administrator's computer can for example check the packet headers by itself or use for this the services of the network firewall on the gateway computer if such a firewall is being used, or use some other routing information to know from which computers the data is coming. This feature is very important and can be used also independently of other features to find cases where the actual amount of data sent-out does not fit the amount reported, even for example the amount reported by the dialer of the operating system. Of course, more than one administrator can exist. Another possible variation is to use for example for each computer (or each group of computers) a modem or network card or other communications device (111 in Fig 1) capable of monitoring at least the amounts of data sent out so that this communication device can preferably report back to the Security System of the computer how much data actually went out, so that preferably the communications can be immediately blocked and an alert issued if the amount of actual communication does not fit the amount reported by the Security System of that computer. (Preferably the blocking in this case is done by the Security system, however another possible variation is that it can be done also by the communications device itself, or by the administrator's computer or the gateway computer in organizations where all the traffic goes through them). This has the advantage that it can be used also for single computers or small groups of computers that don't use central control, however it could be used also in organizations with central control. Again, this is important and can be used also independently of the other features to find cases where the actual amount of data sent-out does not fit the amount reported, even for example the amount reported by the dialer of the operating system. Another possible variation of this is for example reporting similarly also how much data went in. Another possible variation in organizations is that reports about such cases of incongruities are reported automatically by these communication devices also (or instead) for example to the system administrator.

Fig. 2 shows a preferred method for monitoring, checking and authorizing access to hooked functions that are called due to a disk related action (201) (such as file open, file read, file write, file change, disk read, disk write, disk format, etc.). Preferably the function is tunneled to the proper method of access (202) (read, write, query, etc.). Then the Security System retrieves caller's identity (203), retrieves its relevant information from the database (700), if needed, and retrieves the required file action parameters (204) (such as file name, path name, etc.). The parameters are tracked (211) and, if needed, relevant parts are stored in database (700) for further use (for example for statistics). If needed, an access to rules settings in the database (700) is made to check whether the current action is permitted, and the answer's origin is authorized to prevent hacking of the Security System (207). (For example, if the program tries to access a file within its own natural environment, it is not necessary to access the database). Also, preferably the Security System can take into consideration also if the action was initiated by the user or by the application, as described in Fig. 1b. If hacking was spotted, the Security System preferably proceeds to special termination process (1001). If origin of answer is authenticated as coming indeed from the database, the Security System performs a check whether the action is permitted. If not, the Security System can for example ask permission from the user, or terminate the process, or tell it that something does not exist, or tell it that the request has been done (without actually doing it), or do the above things if the user has not agreed, or choose other actions, preferably depending also on the amount of visibility wanted by the user (1002), and if authorized it passes on the parameters to the original hooked function (212), and, if needed, the database is updated with the new authorization. Also, it should be noted that this and the other figures, and especially the flowcharts are just general examples, and various steps can for example change or be in a different order.

Fig. 3 shows a preferred method for monitoring, checking and authorizing access to hooked functions that are called due to a communication related action (301) (such as open connection, close connection, send, receive, etc.). Preferably, the function is tunneled to the proper method of access (302) (send, receive, etc.). Then the Security System retrieves caller's identity (303), retrieves its relevant information from database (700) and retrieves required communication action parameters (304) (such as handle id, address, protocol, etc.). The parameters are tracked (311) and, if needed, relevant parts are stored in database (700) for further use (for example for statistics). Also, preferably, when possible, the Security System can take into consideration also if the action was initiated by the user or by the application, as described in Fig. 1b. If needed, an access to rules settings in the database (700) is made to check whether the current action is permitted and the answer's origin is authorized to prevent hacking of the Security system (307). If hacking was spotted the Security System preferably proceed to special termination process (1001). If origin of answer is authenticated as coming indeed from the database, the Security System performs a check whether the action is permitted. If not, the Security System can for example ask permission from the user, or terminate the process, or tell it that something does not exist, or tell it that the request has been done (without actually doing it), or do the above

things if the user has not agreed, or choose other actions, preferably depending also on the amount of visibility wanted by the user (1002), and if authorized it passes on the parameters to the original hooked function (312), and, if needed, the database is updated with the new authorization.

Fig. 4 shows a preferred method for monitoring, checking and authorizing access to hooked functions that are called due to a registry related action (401) (such as read, write, change, etc.). Preferably, the function is tunneled to the proper method of access (402) (read, write, etc.). Then the Security System retrieves caller's identity (403), retrieves its relevant information from database (700) and required registry action parameters (404) (such as key, value, etc.). The parameters are tracked (411) and, if needed, relevant parts are stored in database (700) for further use (for example for statistics).. An access to rules settings in the database (700) is made to check whether the current action is permitted, answer's origin is authorized to prevent hacking of the Security System (407). If hacking was spotted the Security system preferably proceeds to special termination process (1001). If origin of answer is authenticated as coming indeed from the database, the Security System performs a check whether the action is permitted. If not, the Security System can for example ask permission from the user, or terminate the process, or tell it that something does not exist, or tell it that the request has been done (without actually doing it), or do the above things if the user has not agreed, or choose other actions, preferably depending also on the amount of visibility wanted by the user (1002), and if authorized it passes on the parameters to the original hooked function (412) and, if needed, the database is updated with the new authorization.

Fig. 5 shows what preferably happens when any executable files are being loaded for execution (501) by the operating system. The Security System is notified about it and checks it before it actually starts running. Furthermore, the file is being accessed in an earlier phase (see fig. 2) when the Security System permits the access to the file (for example, if format.exe was denied for all it won't reach this phase) as it is being accessed before loading into memory (see fig. 2). The Security System tracks file parameters and relevant data (502) (such as process id (PID), threads, allocated memory, etc.) for further use, stores them in the database (700) if needed, and passes on the parameters.

Fig. 6 shows a preferred method for monitoring, checking and authorizing access to hooked functions that are called due to due to a memory related action (601) (such as read, write, etc.). Then the Security System retrieves caller's identity (602), retrieves its relevant information from database (700), gets its parts (libraries, etc.) and its self-allocated memory (physical, virtual, etc.) (603), and checks if the process exceeded its memory borders (604). If it exceeded it, the Security System can for example ask permission from the user, or terminate the process, or tell it that something does not exist, or tell it that the request has been done (without actually doing it), or do the above things if the user has not agreed, or choose other actions, preferably depending also on the amount of visibility wanted by the user (1002), otherwise it passes on the parameters to the original hooked function (605). This feature is implemented to the extent possible since its implementation may be limited or partially limited on

various operating systems. The optional additional hardware described in fig. 8 might also be useful in this context if needed.

Referring to fig. 7, we show preferred main parts and methods of a Security System database. The database or parts of it are located in computer's memory and in storage media. Any access to the database is encrypted (701) and its origin identified (702). The authentication is checked (703) and if hacking was spotted the program preferably proceeds to special termination process (1001). If the access is authenticated the database may set or retrieve information (704) from or to the records (740) which preferably contain statistics records (751), Process ID (PID) records (752), additional records (760), log of activity (770) and Security rules (740) which preferably contain info such as file records (741), Network records (742) and Registry records (743). Each group of the rule records preferably contains the following information: acquired user's rules, pre-distribution acquires rules, default rules and variant parameters (as described above). If the request is for storing information, the request is performed and returned to caller (706) (one of the Security System inner functions). If the request is for retrieving information, the following preferably occurs: The database keeps track of statistics and analyzes (707). If the Security System spots any suspicious deviation in activity, the answer returned to the caller function is negative and the appropriate explanation passed through (710) (this action is performed when handling information that is not inner security database such as PID-752, etc.), otherwise it returns the answer that was retrieved from the database (709).

Referring to Fig. 8, another possible variation is that the Security System may also include an optional hardware element (800) which gathers (804) and/or logs (805) monitored hardware port accesses (803), DMA (801), IRQ (802), etc. The monitoring hardware mainly monitors access to storage devices (especially hard disk controller) and access to network devices (such as modem, network cards, etc.). Preferably, the monitoring hardware has an interface (811) for transfer of information from the Security System's software (104) to said hardware element (800) (such as through accessing read and/or write ports in said hardware element (800)) and for immediate feedback to the Security System's software (104) (such as through accessing read and/or write ports in said hardware element (800), through interrupts, etc.) so that it can alert the Security System's software (104) to any events that have been defined in the built-in local database (806). The comparison of events between the software monitoring and the hardware monitoring can preferably be done by either the hardware element (800), by the software part of the Security System (104) or by both. When either the hardware element (800) or the Security System's software decides that unknown access has been made to the above monitored hardware without apparent corresponding event on the top system level as monitored by the Security System software (104), the event is intercepted and reported. Monitoring and catching these events enables the Security System to further close any remaining loopholes in the operating system and programs that may override agents hooking, and ultimately even catch and intercept even backdoors in the operating system if they exist.

Referring to Fig. 9, we show an overview of a preferred self-preservation method. Any Security System part that is being called (901) performs a regular check every defined time (902) for all Security System files integrity (903) and its running functions' (as described in fig. 1) integrity (904). If a deviation is found (905), it informs the user for full understanding of the situation and performs a Self-preservation interception and report (1001). In addition to this, in order to protect itself in memory, preferably the security system defines a part of the physical memory so that no other process can access it except by using a limited number of calling gates (such as when calling one of the hooked functions), and any other attempt to access this memory area for example for reading or writing causes a CPU exception which transfers control to the Security System. Since the Security system can know from this which application tried to "attack" it, the security system preferably initiates "anti-hacking" measures, such as for example disabling the attacking part of the process, terminating the process, destroying the process's environment, etc.

Referring to Fig. 10, we show a preferred method of the interception process. It preferably contains two major interception routes: The first is a normal interception (1002) - it is used when an executable tries to perform an unauthorized activity. In that case it preferably notifies the user (1101) (as described above), blocks the parameters from reaching the original function (1006), and can for example inform the original caller (the program that requested the function) about function failure. The second is a Self-preservation interception (1001). It is used when the Security System detects an intrusion of any kind by an offensive program or a hacker. In that case preferably it terminates the offensive program immediately (1007) (such as unload from memory, etc.) (Method of termination may be different from operating system to another), and the Database (700) is modified so it marks the offensive program and/or its files accordingly (1009) (such as not allowing the access to them, etc.). A self-check is being performed (900) (as described in fig. 9) and if the Security System is endangered (1010), it starts Survival emergency procedures (1011) (such as reinstall, shutdown parts, reload, etc.). If not, it continues monitoring (1010). Although it may seem from the diagram that in certain cases there might occur endless loops, this is not the case in reality, it only seems so because the diagram is simplified.

Referring to Fig. 11, we show a graphic illustration of a preferable way in which processes may be segregated and controlled. Whenever a process (1111) attempts to access other processes or their natural environments (1113) or possibly important system resources (1114-1124), it has to go through the Security System's interception and decision engine, so that practically a virtual environment or virtual computer (1112) is created around it. However, it should be noted that this graphic illustration is just a possible example. Not all of these functions are necessarily implemented. (Category 1122 – other – refers to other possible resources that may be relevant for example in other operating systems or other CPUs). A more extreme possible implementation of this concept (as illustrated also in Fig. 12) is for example that every time a new program is installed, it is given the illusion that nothing else exists on the computer but itself and the operating system and the computer resources that it

is allowed to see, so that only the user and certain programs such as for example the relevant parts of the Windows directory explorer and privileged programs given special explicit permission by the user or by predefined rules can see the real directory structure and resources. These permissions can be given either at the level of an entire program, or for example at the level of segments or parts of the program, so for example the part of explorer that deals with the screen does not need to be given permission to access the directory system. This can further limit for example the extent of damage that can be caused by various exploits. This way, it is like an infinite set of new installed computers, each with a clean and new operating system, and each computer runs only one program: For example, one runs Internet Explorer, a second runs Windows Word, a third runs DOOM II, and so on. For this implementation preferably the Security System is the first thing installed after the operating system, and the security system preferably relies mainly on identifying if the user or the program initiated each security-sensitive action in order to decide automatically if to allow it or not. In order to make this more efficient in organizations, preferably one computer can be used for example for learning all of the segregation rules and virtual environment parameters for each program, and this knowledge can be transferred to all the other computers in the organization, without the need to install the Security System before the other applications in the other computers.

Referring to Fig. 12, we show a visual illustration of a more extreme implementation of keeping each program in a 'Bubble' of virtual environment, so that the application can only see itself (2001) and not other programs except its virtual environment (2002), which contains the operation system and the resources it is allowed to see. Only by explicit permission from the user can the program see other programs or their data or access other system resources.

Referring to Fig. 13, we show a visual illustration of a preferable configuration in a possible variation in which individual computers in an organization (3001-3005), each with its own installation of the Security System, are connected to the Internet (3020) through the central authority's computer, such as for example the system administrator (3010) (or though another gateway computer which supplies information to the central authority about the amount of data actually sent from each computer), with it's own installation of the Security System, so that the Security System on the central authority's computer can also notice and intercept communication attempts from computers where the amount of actual communication does not fit the amount reported by the Security System of that computer, as described in the reference to fig. 1b.

While the invention has been described with respect to a limited number of embodiments, it will be appreciated that many variations, modifications, expansions and other applications of the invention may be made which are included within the scope of the present invention, as would be obvious to those skilled in the art.

CLAIMS

We claim:

1. A security system for computers, capable of creating automatic segregation between programs.

2. The system for claim 1, capable of monitoring and learning the reasonable behavior of the operating system and software applications in a computer system and enforcing generic security rules in order to reduce to the chance of the users of said computer being cheated by malicious software applications, said system being comprised of:
    a. A monitoring and capturing system;
    b. A database of security rules, comprising at least: a set of default rules, a set of pre-distribution acquired rules that are good for many users of the selected operating system, and acquired additional user-defined rules;
    c. A user interface, which can interact with the user in order to learn acceptable behavior patterns, warn the user of perceived dangers and wait for his authorization whenever necessary.

3. The system of claim 2, wherein said main 3 elements are:
    a. A monitoring and capturing system, which constantly monitors the security-sensitive elements of the computer system, and mainly all relevant peripheral device activities, and especially storage devices and communication devices, and detects and selectively intercepts any suspicious and dangerous behaviors and acts upon it in accordance with default and acquired sets of security rules, and may ask for user authorization and guidance whenever necessary; the files related to said system being constantly monitored as a high-security protected area;
    b. A database of security rules, comprising at least: a set of default rules, a set of pre-distribution acquired rules that are good for most users of the selected operating system, and acquired additional user-defined rules, said database being constantly monitored as a high-security protected area;
    c. A user interface, which can interact with the user for at least: learning acceptable behavior patterns, warning the user of perceived dangers and waiting for his authorization whenever necessary, and allowing the user to view and modify the database of authorizations.

4. The system of claim 3 wherein said user interface at least also warns the user more explicitly in cases of potentially highly dangerous activities.

5. The system of claim 3 wherein said database comprises also of at least continuously learned statistics of normal and reasonable behavior of programs in the user's computer.

6. The system of claim 3 wherein said user interface at least also allows the user to view statistics of behavior of important programs and especially programs that are allowed to access communication channels, especially in what is related to sending and receiving data over the communication lines.

7. The system of claim 3 wherein said database comprises also of at least a log of the questions that the Security System asked the user and his replies kept at least for a certain period.

8. The system of claim 3 wherein said database comprises also of at least, when needed, a log of suspicious activities detected kept at least for a certain period.

9. The system of claim 3 wherein said security rules and functions performed by the Security System comprise at least the following functions:

    a. Constantly monitoring the security-sensitive elements of the computer system, and mainly all relevant peripheral device activities, and especially storage devices and communication devices, and detecting and selectively intercepting suspicious behaviors and dangerous behaviors and acting upon them in according with default and acquired sets of security rules,
    b. Default automatic segregation of programs into their natural environments,
    c. Warning the user and request for authorization for security-sensitive activities and especially any first-time attempts to access communication channels,
    d. Enabling the user to request immediate interception and warning of the user of any attempts of external programs from the network to connect to the user's computer through the communication channels,
    e. Interception and more explicit warning of the user about potentially highly dangerous activities.

10. The system of claim 9, comprising also at least warning the user about significant statistical deviations from normal behaviors of applications and operating system and especially as relates to suddenly sending out large amounts of data.

11. The system of claim 9, comprising also at least enabling the user to request enforcing of general limitations on the communication ports allowed to be opened and when needed also limitations on types of protocols allowed.

12. The system of claim 9, comprising also at least monitoring and intercepting as much as possible all attempts of applications to gain direct port accesses to security sensitive devices and especially the storage media and the communication channels.

13. The system of claim 9, comprising also at least implementing Virtual Shared data areas on the storage media, such as for temporary files and for accessing keys in the registry, so that programs are given the illusion that they are accessing the shared area, but in reality are each redirected to a separate private area.

14. The system of claim 9, comprising also at least pushing, to the extent possible, at least part of the operating system from the most privileged processor ring to a lower privilege ring and enabling needed functions to run in said lower privilege ring, in order to catch backdoors and loopholes in the operating system itself.

15. The system of claim 9 wherein said monitoring and capturing system includes also a hardware element which monitors hardware accesses on the computer's bus and has a 2-way interface for communication from and to the Security System's software, so that the Security System can discover events where access has been made to the security-sensitive ports, especially the storage media and the communication channels, without an apparent corresponding event on the system level as monitored by said Security System's software.

16. The system of claim 9 wherein said default automatic segregation is implemented so that, by default, each program is allowed to access, such as read, write, execute, create, and delete, files only within its natural environment, which is mainly the directory in which it is installed, its sub-directories, and - for reading only - non-strategic shared files, unless specifically given it more rights.

17. The system of claim 9 wherein said computer's operating system and usage is configured to function mainly as a personal computer.

18. The system of claim 9 wherein said computer's operating system and usage is configured to function mainly as a network server

19. The system of claim 9 wherein said computer is a computerized gadget, such as for example cellular phone, palm pilot, car computer, etc.

20. The system of claim 3 wherein high security protected areas are also encrypted.

21. The system of claim 3 wherein high security protected areas are also automatically backed up to as least one more area for additional safety.

22. The system of claim 20 wherein high security protected areas are also automatically backed up to as least one more area for additional safety

23. The system of claim 3 wherein said communication channels include also USB interfaces.

24. The system of claim 3 wherein said communication channels include also wireless devices, such as for example Bluetooth devices.

25. The system of claim 3 wherein said protocols include also protocols for sending Faxes.

26. The system of claim 9 wherein high security protected areas are also encrypted.

27. The system of claim 9 wherein high security protected areas are also automatically backed up to as least one more area for additional safety.

28. The system of claim 26 wherein high security protected areas are also automatically backed up to as least one more area for additional safety

29. The system of claim 9 wherein said communication channels include also USB interfaces.

30. The system of claim 9 wherein said communication channels include also wireless devices, such as for example Bluetooth devices.

31. The system of claim 9 wherein said protocols include also protocols for sending Faxes.

32. A security method for computers, capable of creating automatic segregation between programs.

33. The method for claim 32, capable of monitoring and learning the reasonable behavior of the operating system and software applications in a computer system and enforcing generic security rules in order to reduce the chance of the users of said computer being cheated by malicious software applications, said method comprising the steps of :
    a. Providing a monitoring and capturing system;
    b. Creating and maintaining a database of security rules, comprising at least: a set of default rules, a set of pre-distribution acquired rules that are good for many users of the selected operating system, and acquired additional user-defined rules;
    c. Providing a user interface, which can interact with the user in order to learn acceptable behavior patterns, warn the user of perceived dangers and wait for his authorization whenever necessary.

34. The method of claim 33, wherein said main 3 elements are:
    a. Providing a monitoring and capturing system, which constantly monitors the security-sensitive elements of the computer system, and mainly all relevant peripheral device activities, and especially storage devices and communication devices, and detects and selectively intercepts any suspicious and dangerous behaviors and acts upon it in accordance with default and acquired sets of security rules, and may ask for user authorization

and guidance whenever necessary; the files related to said system being constantly monitored as a high-security protected area;

b. Creating and maintaining a database of security rules, comprising at least: a set of default rules, a set of pre-distribution acquired rules that are good for most users of the selected operating system, and acquired additional user-defined rules, said database being constantly monitored as a high-security protected area;

c. Providing a user interface, which can interact with the user for at least: learning acceptable behavior patterns, warning the user of perceived dangers and waiting for his authorization whenever necessary, and allowing the user to view and modify the database of authorizations.

35. The method of claim 34 wherein said user interface at least also warns the user more explicitly in cases of potentially highly dangerous activities.

36. The method of claim 34 wherein said database comprises also of at least continuously learned statistics of normal and reasonable behavior of programs in the user's computer.

37. The method of claim 34 wherein said user interface at least also allows the user to view statistics of behavior of important programs and especially programs that are allowed to access communication channels, especially in what is related to sending and receiving data over the communication lines.

38. The method of claim 34 wherein said database comprises also of at least a log of the questions that the Security System asked the user and his replies kept at least for a certain period.

39. The method of claim 34 wherein said database comprises also of at least, when needed, a log of suspicious activities detected kept at least for a certain period.

40. The method of claim 34 wherein said security rules and functions performed by the Security System comprise at least the following functions:

a. Constantly monitoring the security-sensitive elements of the computer system, and mainly all relevant peripheral device activities, and especially storage devices and communication devices, and detecting and selectively intercepting suspicious behaviors and dangerous behaviors and acting upon them in according with default and acquired sets of security rules,

b. Default automatic segregation of programs into their natural environments,

c. Warning the user and request for authorization for security-sensitive activities and especially any first-time attempts to access communication channels,

d. Enabling the user to request immediate interception and warning of the user of any attempts of external programs from the network to connect to the user's computer through the communication channels,
e. Interception and more explicit warning of the user about potentially highly dangerous activities.

41. The method of claim 40, comprising also at least warning the user about significant statistical deviations from normal behaviors of applications and operating system and especially as relates to suddenly sending out large amounts of data.

42. The method of claim 40, comprising also at least enabling the user to request enforcing of general limitations on the communication ports allowed to be opened and when needed also limitations on types of protocols allowed.

43. The method of claim 40, comprising also at least monitoring and intercepting as much as possible all attempts of applications to gain direct port accesses to security sensitive devices and especially the storage media and the communication channels.

44. The method of claim 40, comprising also at least implementing Virtual Shared data areas on the storage media, such as for temporary files and for accessing keys in the registry, so that programs are given the illusion that they are accessing the shared area, but in reality are each redirected to a separate private area.

45. The method of claim 40, comprising also at least pushing, to the extent possible, at least part of the operating system from the most privileged processor ring to a lower privilege ring and enabling needed functions to run in said lower privilege ring, in order to catch backdoors and loopholes in the operating system itself.

46. The method of claim 40 wherein said monitoring and capturing system includes also a hardware element which monitors hardware accesses on the computer's bus and has a 2-way interface for communication from and to the Security System's software, so that the Security System can discover events where access has been made to the security-sensitive ports, especially the storage media and the communication channels, without an apparent corresponding event on the system level as monitored by said Security System's software.

47. The method of claim 40 wherein said default automatic segregation is implemented so that, by default, each program is allowed to access, such as read, write, execute, create, delete, etc., files only within its natural environment, which is mainly the directory in which it is installed, its sub-directories, and - for reading only - non-strategic shared files, unless specifically given more rights.

48. The method of claim 40 wherein said computer's operating system and usage is configured to function mainly as a personal computer.

49. The method of claim 40 wherein said computer's operating system and usage is configured to function mainly as a network server

50. The method of claim 40 wherein said computer is a computerized gadget, such as for example cellular phone, palm pilot, car computer, etc.

51. The method of claim 34 wherein high security protected areas are also encrypted.

52. The method of claim 34 wherein high security protected areas are also automatically backed up to as least one more area for additional safety.

53. The method of claim 51 wherein high security protected areas are also automatically backed up to as least one more area for additional safety

54. The method of claim 34 wherein said communication channels include also USB interfaces.

55. The method of claim 34 wherein said communication channels include also wireless devices, such as for example Bluetooth devices.

56. The method of claim 34 wherein said protocols include also protocols for sending Faxes.

57. The method of claim 40 wherein high security protected areas are also encrypted.

58. The method of claim 40 wherein high security protected areas are also automatically backed up to as least one more area for additional safety.

59. The method of claim 57 wherein high security protected areas are also automatically backed up to as least one more area for additional safety

60. The method of claim 40 wherein said communication channels include also USB interfaces.

61. The method of claim 40 wherein said communication channels include also wireless devices, such as for example Bluetooth devices.

62. The method of claim 40 wherein said protocols include also protocols for sending Faxes.

63. A computer security system capable of automatic segregation of programs into their natural environments so that each program is allowed to access, such as read, write, execute, create, delete, etc., files only within its natural environment, which is mainly the directory in which it is installed,

its sub-directories, and - for reading only - non-strategic shared files, unless specifically given more rights.

64. A method of implementing security in computers by automatic segregation of programs into their natural environments so that each program is allowed to access, such as read, write, execute, create, delete, etc., files only within its natural environment, which is mainly the directory in which it is installed, its sub-directories, and - for reading only - non-strategic shared files, unless specifically given more rights.

65. A security system in computerized gadgets, such as for example cellular phone, car computer, etc., wherein access to highly sensitive data, such as credit card details or private encryption keys, needs explicit permission by the user.

66. A security system in computerized gadgets, such as for example cellular phone, car computer, etc., wherein any attempt to automatically generate an outgoing communication needs explicit permission by the user.

67. The system of claim 64 wherein any attempt to automatically generate an outgoing communication needs explicit permission by the user.

68. The system of claim 64 wherein any attempts to alter sensitive data, such as for example EMROMM and important system files, need explicit permission by the user.

69. The system of claim 3 wherein the user is an individual.

70. The system of claim 3 wherein the user is an organization and at least some of the control over authorizations is in the hands of at least one central authority, such as the system administrator.

71. The system of claim 70 wherein the Security System of the central authority also automatically checks at least once in a while if the /Security System is functioning properly on the other computers.

72. The system of claim 70 wherein the Security System on the central authority's computer can also notice and intercept communication attempts from computers where the amount of actual communication does not fit the amount reported by the Security System of that computer.

73. The system of claim 71 wherein the Security System on the central authority's computer can also notice and intercept communication attempts from computers where the amount of actual communication does not fit the amount reported by the Security System of that computer.

74. The system of claim 3 wherein the communications device of each computer can also notice and at least report back to the computer about cases where the amount of actual communication does not fit the amount reported by the Security System of that computer.

75. The system of claim 70 wherein the communications device of each computer can also notice and at least report back to the computer about cases where the amount of actual communication does not fit the amount reported by the Security System of that computer.

76. The system of claim 70 wherein the communications device of each computer can also notice and at least report back to the central control about cases where the amount of actual communication does not fit the amount reported by the Security System of that computer.

77. The system of claim 70 wherein the communications device of each group of computers can also notice and at least report back at least to the relevant computer about cases where the amount of actual communication does not fit the amount reported by the Security System of that computer.

78. The system of claim 70 wherein the communications device of each group of computers can also notice and at least report back at least to the central control about cases where the amount of actual communication does not fit the amount reported by the Security System of that computer.

79. The system of claim 3 wherein by default each program can only see itself and the operating system and the resources (software & hardware) that it is allowed to see.

80. The system of claim 9 wherein by default each program can only see itself and the operating system and the resources (software & hardware) that it is allowed to see.

81. The system of claim 3 wherein the Security System also identifies if the user or the application initiated a potential security-risk command, such as for example accessing a file outside the natural environment of the program for a program that still does not have that privilege, and so can allow more flexibility and less limitations if the command was initiated directly by the user than if it was initiated by the application.

82. The system of claim 9 wherein the Security System also identifies if the user or the application initiated a potential security-risk command, such as for example accessing a file outside the natural environment of the program for a program that still does not have that privilege, and so can allow more flexibility and less limitations if the command was initiated directly by the user than if it was initiated by the application.

83. The system of claim 81 wherein the Security System also makes sure that programs cannot create the false impression that certain actions were initiated by the user by falsifying user input through one of the input devices.

84. The system of claim 82 wherein the Security System also makes sure that programs cannot create the false impression that certain actions were initiated by the user by falsifying user input through one of the input devices.

85. The system of claim 3 wherein the Security System also makes sure that when it requests authorization no other programs can enter false answers as if they were entered by the user through one of the input devices.

86. The system of claim 9 wherein the Security System also makes sure that when it requests authorization no other programs can enter false answers as if they were entered by the user through one of the input devices.

87. The system of claim 3 wherein in the cases where private keys are generated or stored by the browsers, additional rules are used in order to identify the directories where these keys are held, since otherwise accessing the keys by the browser would be within the browser's default authorization.

88. The system of claim 9 wherein in the cases where private keys are generated or stored by the browsers, additional rules are used in order to identify the directories where these keys are held, since otherwise accessing the keys by the browser would be within the browser's default authorization.

89. The method of claim 34 wherein the user is an organization and at least some of the control over authorizations is in the hands of at least one central authority, such as the system administrator.

90. The method of claim 89 wherein the Security System on the central authority's computer can also notice and intercept communication attempts from computers where the amount of actual communication does not fit the amount reported by the Security System of that computer.

91. The method of claim 34 wherein the communications device of each computer can also notice and at least report back to the computer about cases where the amount of actual communication does not fit the amount reported by the Security System of that computer.

92. The method of claim 89 wherein the communications device of each computer can also notice and at least report back to the central control about cases where the amount of actual communication does not fit the amount reported by the Security System of that computer.

93. A security system wherein the user is an organization and at least some of the control over authorizations is in the hands of at least one central authority, such as the system administrator and Security System on the central authority's computer can also notice and intercept communication attempts from computers where the amount of actual communication does not fit the amount reported by the software of that computer.

94. A security method wherein the user is an organization and at least some of the control over authorizations is in the hands of at least one central authority, such as the system administrator and Security System on the central authority's computer can also notice and intercept communication attempts

from computers where the amount of actual communication does not fit the amount reported by the operating system of that computer.

95. A security system wherein the communications device of each computer can also notice and at least report back to the computer about cases where the amount of actual communication does not fit the amount reported by the software of that computer.

96. A security system wherein the user is an organization and the communications device of each computer can also notice and at least report back to the central control about cases where the amount of actual communication does not fit the amount reported by the software of that computer.

97. A security method wherein the communications device of each computer can also notice and at least report back to the computer about cases where the amount of actual communication does not fit the amount reported by the software of that computer.

98. A security method wherein the user is an organization and the communications device of each computer can also notice and at least report back to the central control about cases where the amount of actual communication does not fit the amount reported by the software of that computer.

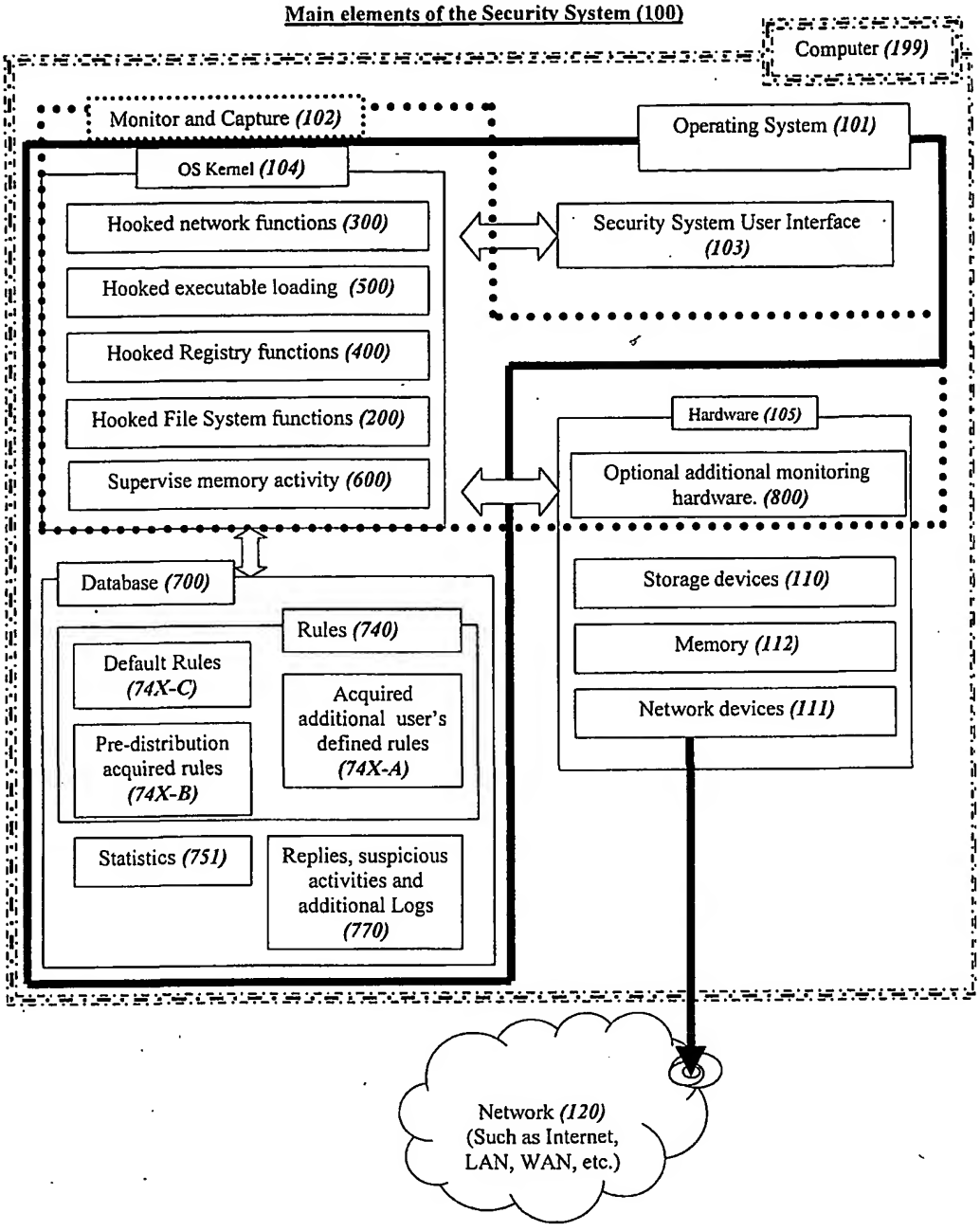

Fig. 1

Interaction between Security System parts overview *(1100)*

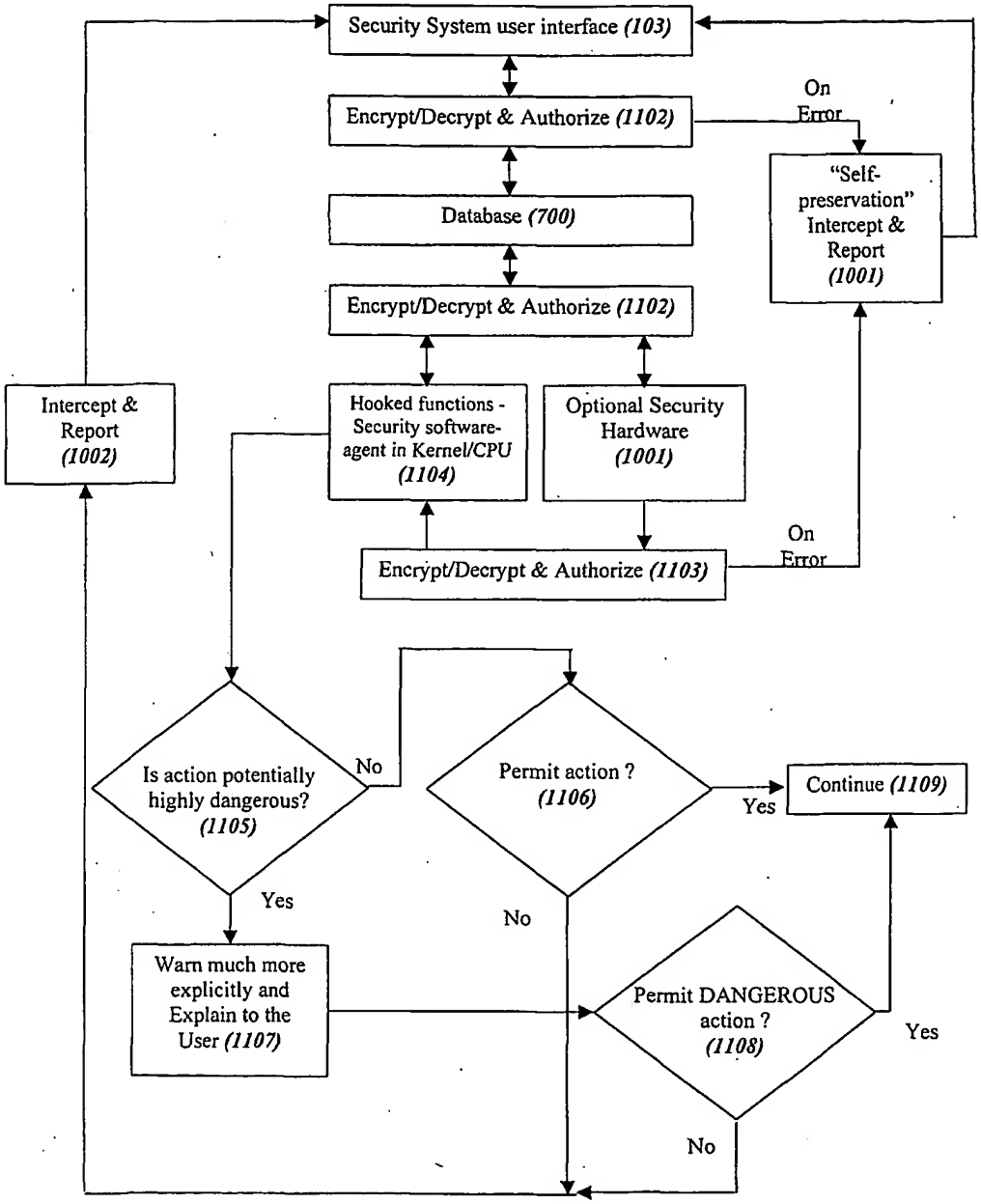

Fig. 1b

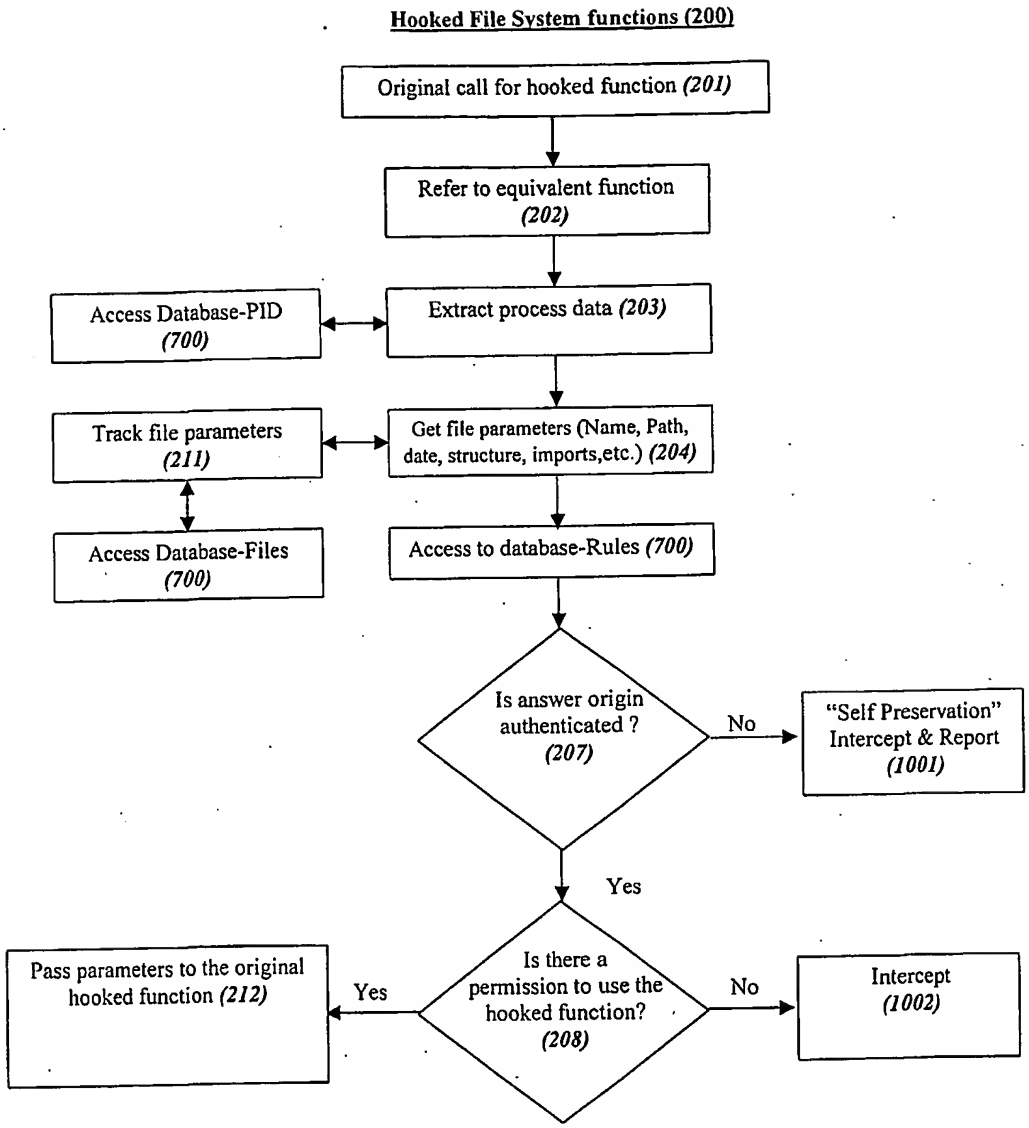

Fig. 2

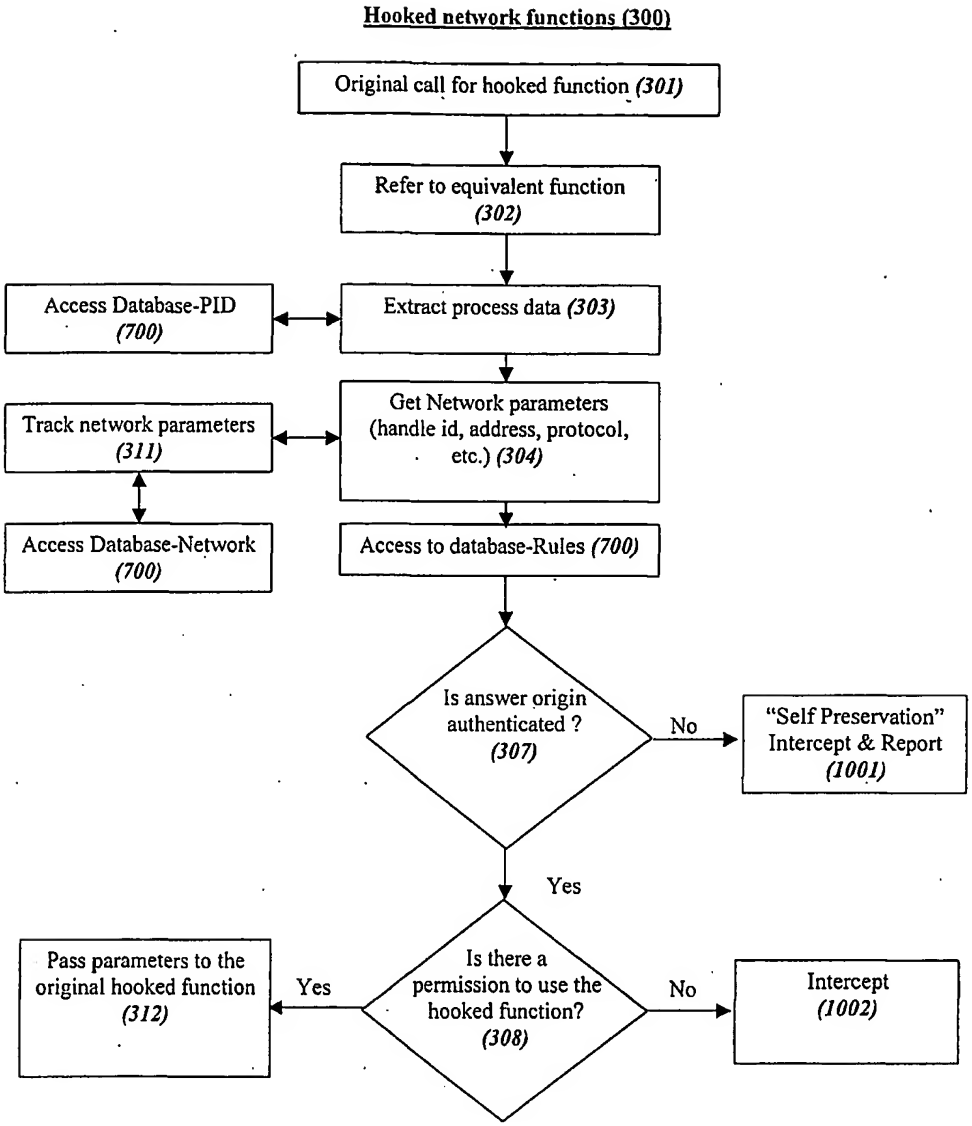

Fig. 3

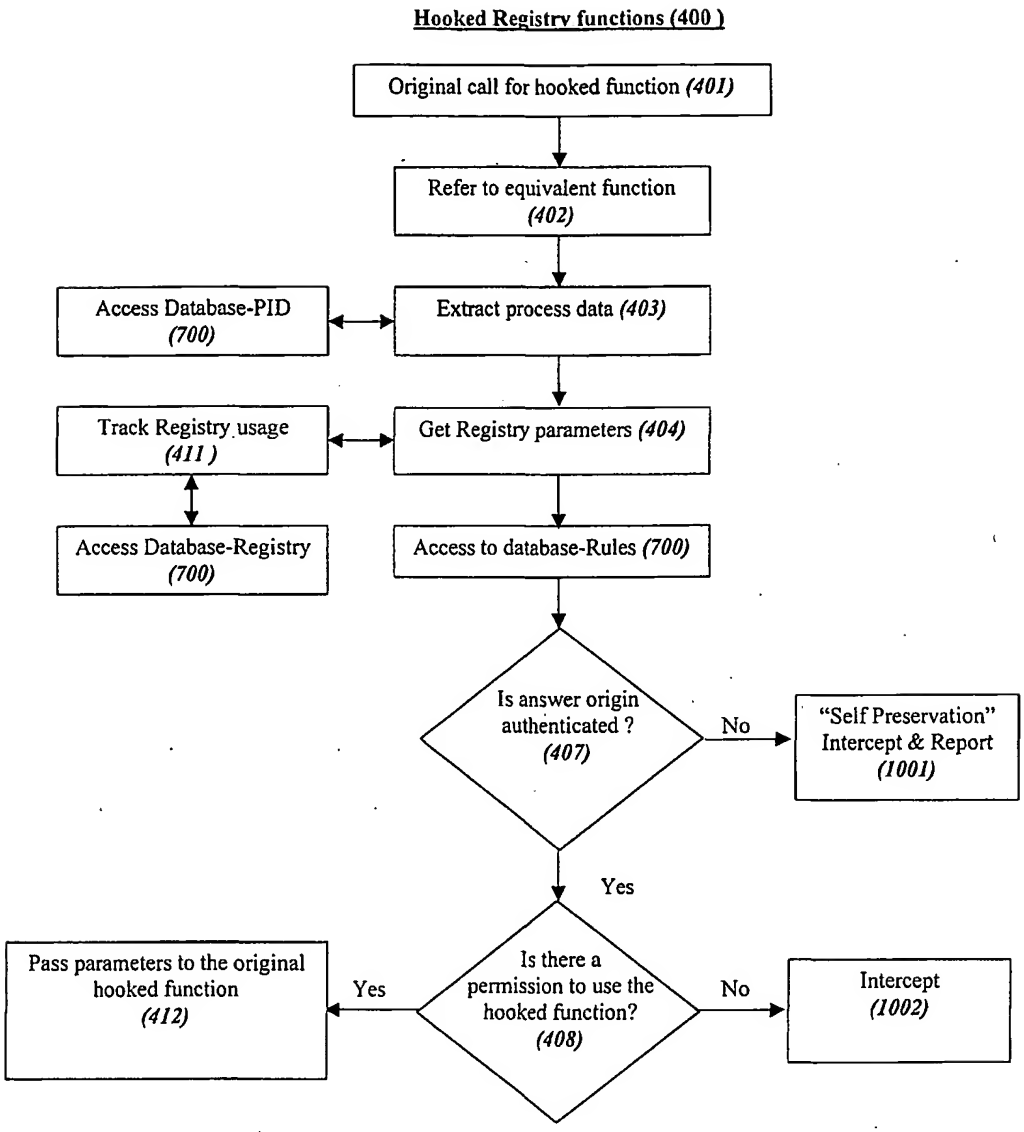

Fig. 4

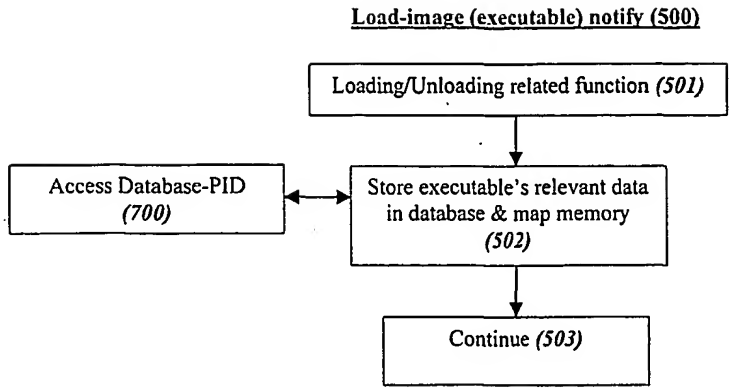

Fig. 5

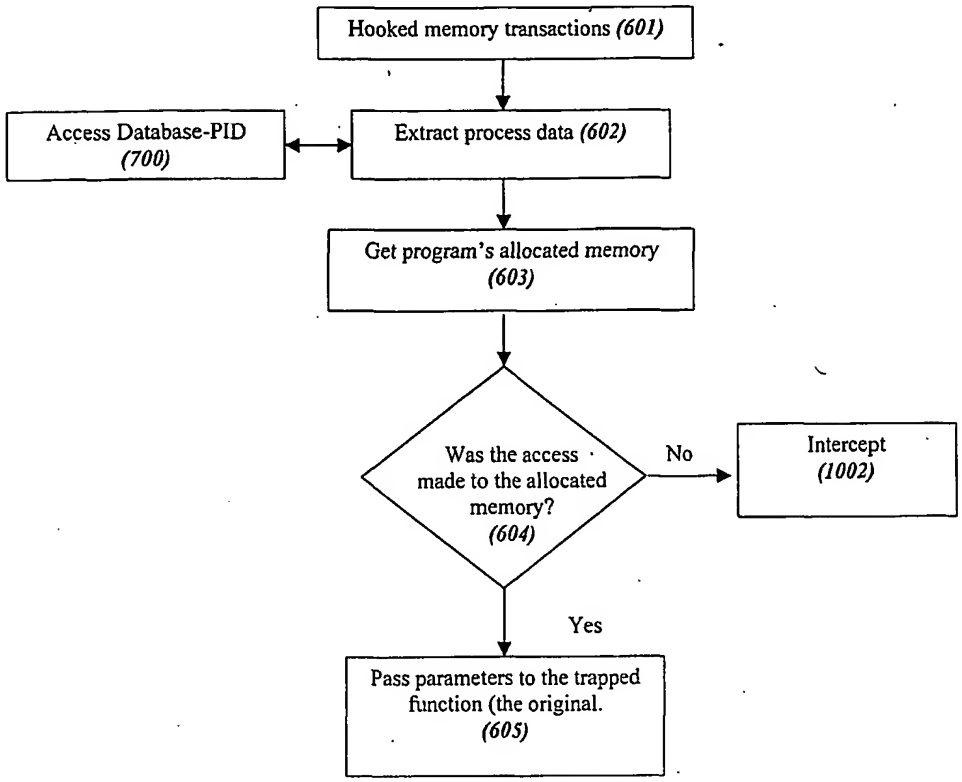

Fig. 6

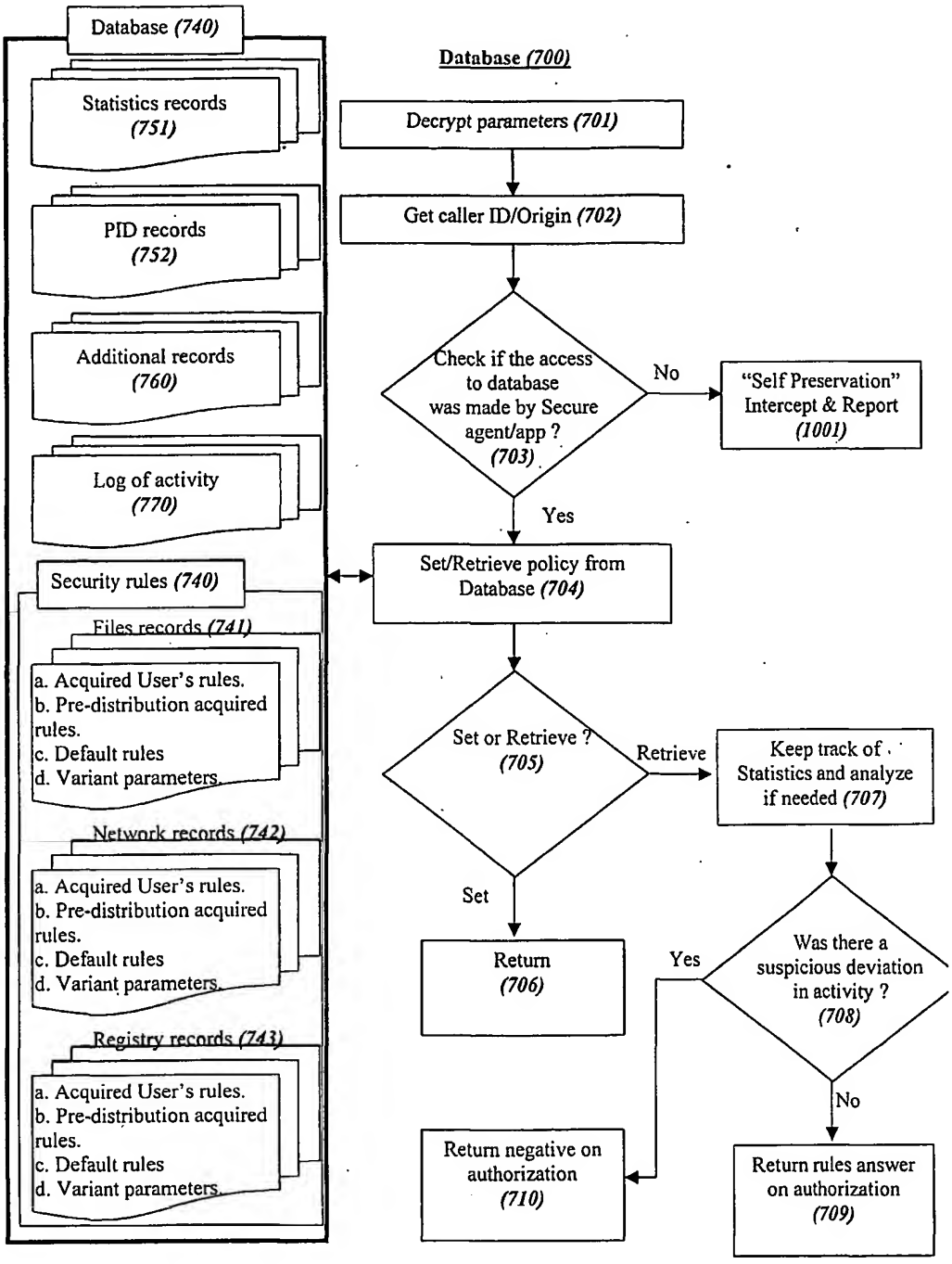

Fig. 7

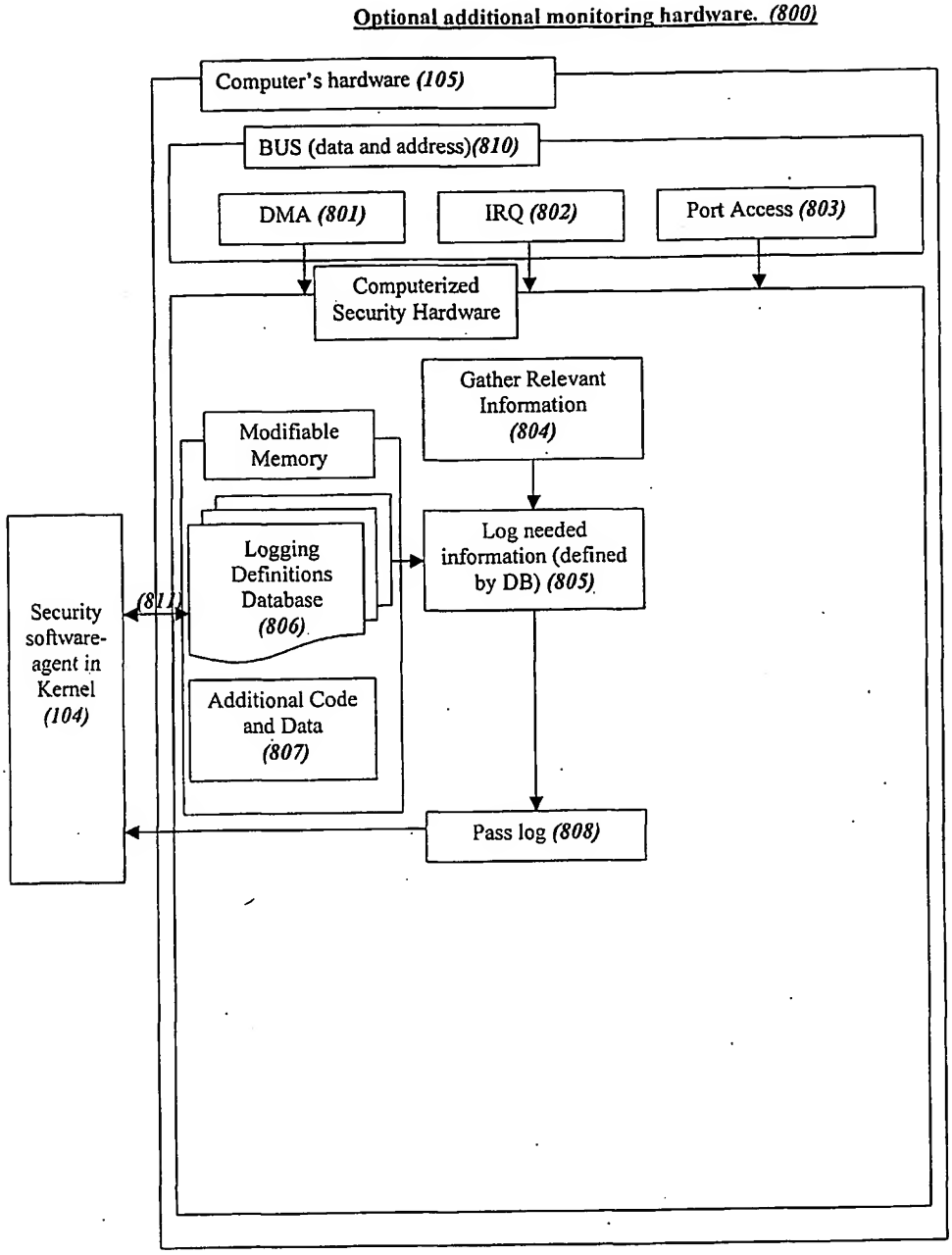

Fig. 8

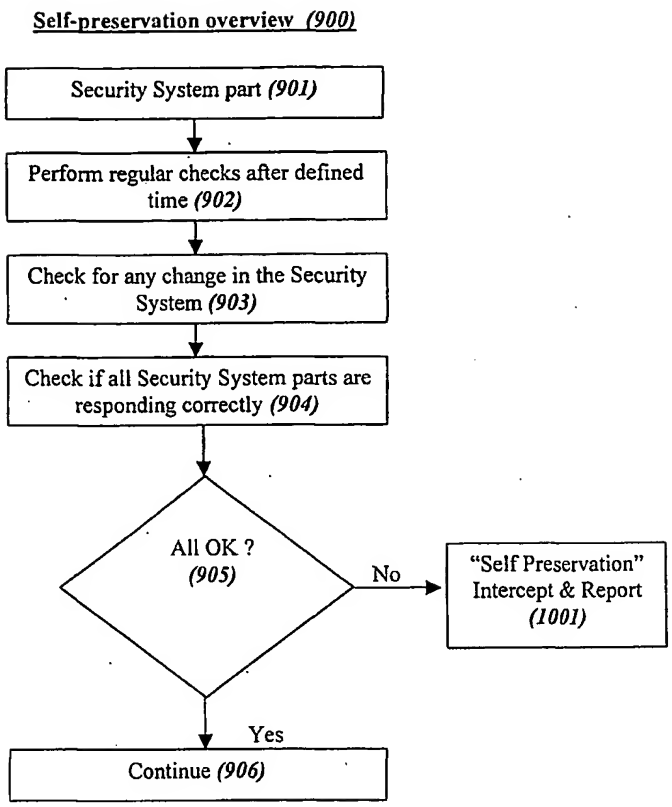

Fig. 9

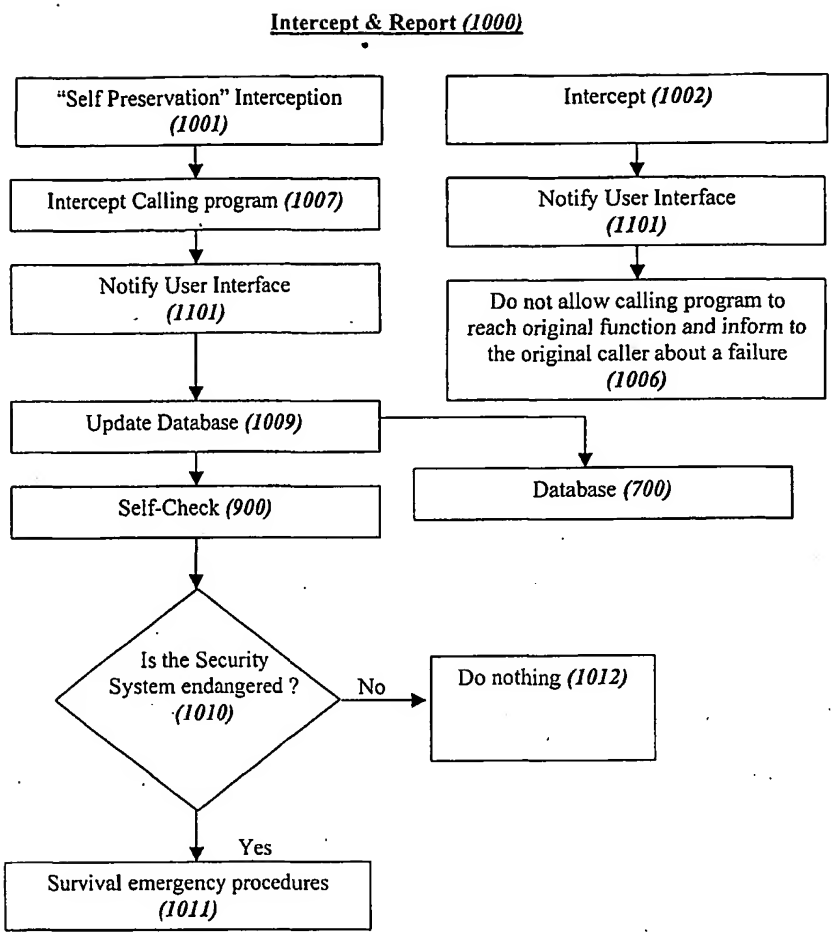

Fig. 10

Fig. 11

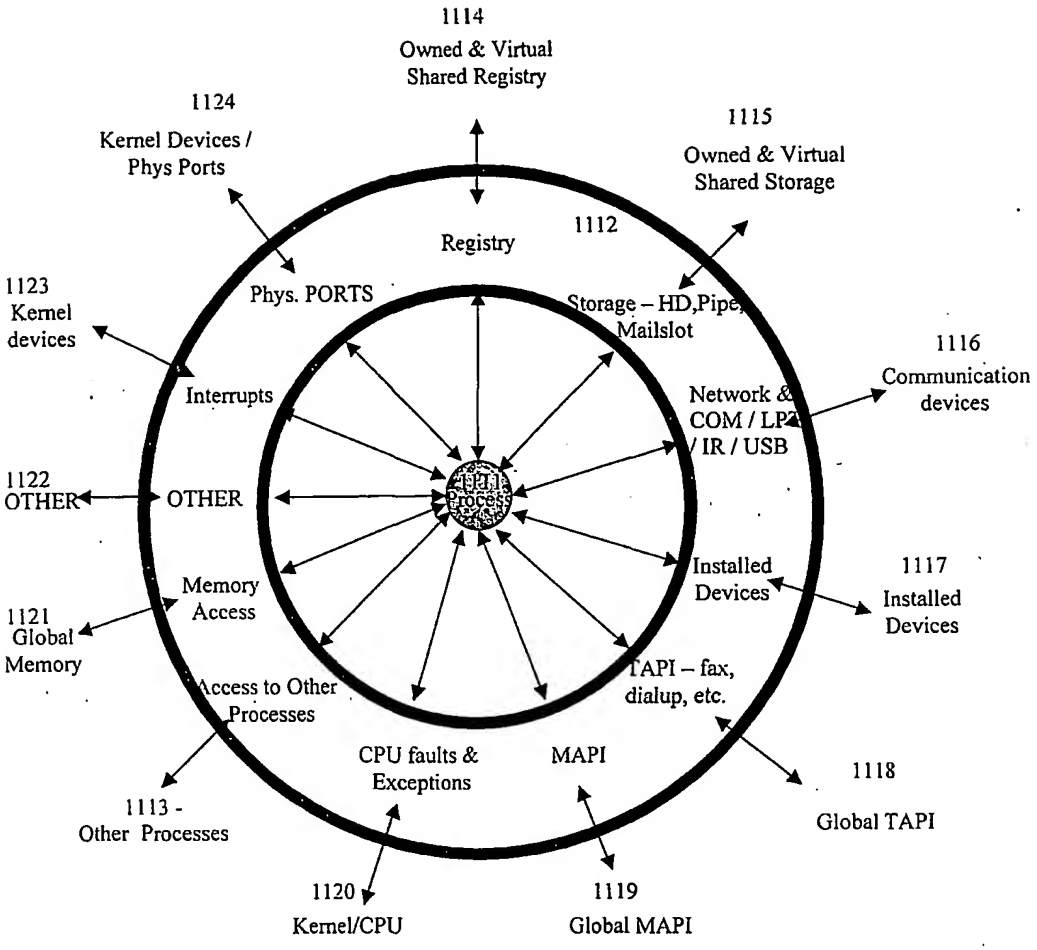

Fig. 12

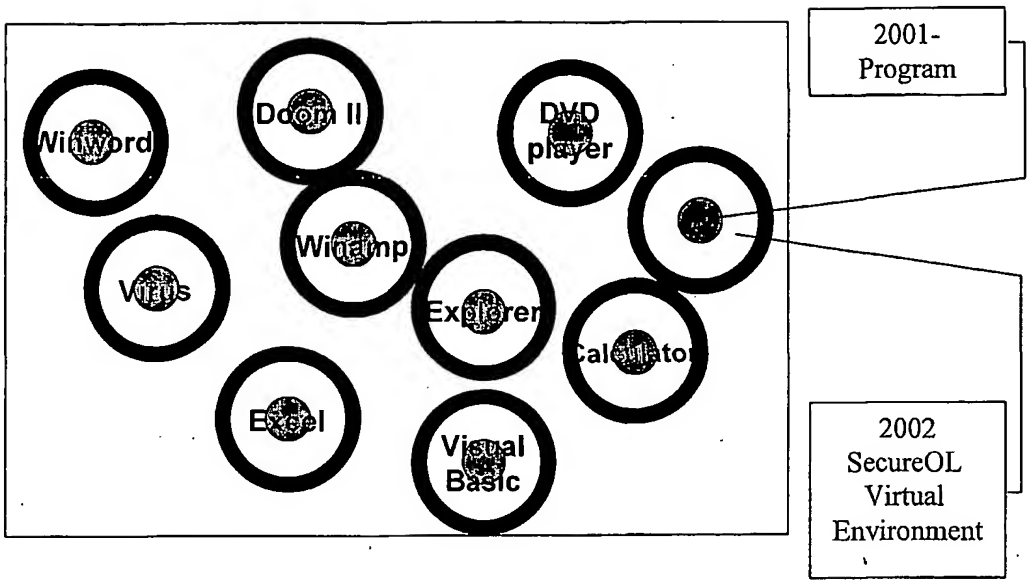

Fig. 13